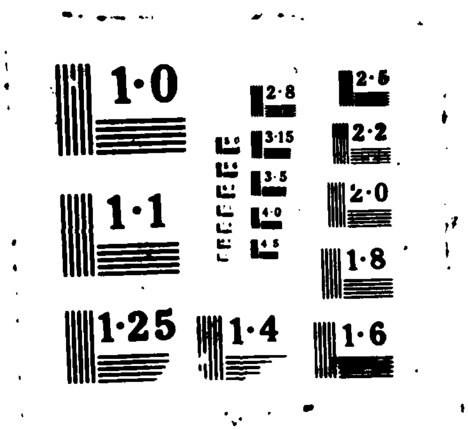
AD-A178 545 AN ALGORITHM FOR THE COMPUTATION OF GENERALIZED LIKELIHOOD OR SELF-CRITIC..(U) RENSSELAUER POLYTECHNIC INST TROY NY SCHOOL OF MANAGEMENT T A DELAHANTY ET AL. 1/1

UNCLASSIFIED 1986 WP-37-86-P26 ARO-18072.20-MA F/G 9/2 NL


1·0
1·1
1·25
1·4
1·6
1·8
2·0
2·2
2·5
2·8
3·15
3·5
4·0
4·5

Rensselaer Polytechnic Institute

School of Management

Working Paper No. 37-86-P26

AN ALGORITHM FOR THE COMPUTATION OF GENERALIZED LIKELIHOOD OR SELF-CRITICAL ESTIMATORS FOR BINARY DATA

by

T. A. Delahanty
and
A. S. Paulson

School of Management
Rensselaer Polytechnic Institute
Troy, NY 12180-3590
(518)266-6586

An Algorithm for the Computation of Generalized Likelihood or Self-Critical Estimators for Binary Data

by

T.A. Delehanty*

and

A.S. Paulson*
Rensselaer Polytechnic Institute

* Research sponsored in part by U.S. Army Research Office Contract DAAG29-81-K-0110.

<u>Documentation for Self-Critical Binary Program</u>

1. Purpose
2. Specficiation (FORTRAN)
3. Description
4. Numerical Method
5. Parameters
6. Error Indicators and Warnings
7. Auxiliary Routines
8. References
9. Storage
10. Precision, Machine Dependent Constants
11. Further Comments
12. Example

1. Purpose

Subroutine BINARY is an implementation of the self-critical estimation procedure of Paulson, Presser and Lawrence (1983). The logistic, Gaussian or Type I extreme value distribution may be selected as tolerance distribution. Estimates are expressed in location-scale form on entry and exit, but results may be printed out in regression form, location-scale form, or in both forms (see Section 3 for a description of the two parametrizations). It is possible to hold location parameters constant during the estimation procedure. Estimation is accomplished by a Newton-Raphson method.

2. Specification (FORTRAN)

```
SUBROUTINE BINARY (N, IX, X, IA, NPAR, ISUB, ISTART, IDEP, IDIST, C,
RELTOL, ABSTOL, MAXIT, IPRINT, IFLAG, BETA, XLOGL, ICOV, COV, LMEM,
MEMORY, IFAULT)
```

3. Description

The routine is applicable to the following modeling situation: For $i = 1, \ldots, n$ (n is the sample size), let v_i be the stress variable, a_i a zero-one indicator of withstand or failure, and $X_i = (x_{1i}, \ldots, x_{pi})^T$ a column vector of covariates. A constant is incorporated as a covariate identically equal to unity. The case $p=0$ is possible, but should be rare.

The tolerance distribution and density, $F(v_i)$ and $f(v_i)$ respectively, depend on the covariates, a scale parameter σ, and a vector of location parameters $\beta = (\beta_1, \ldots, \beta_p)^T$ as follows, where

$$u_i = \frac{(v_i - \beta' X_i)}{\sigma};$$

- logistic, $$F(v_i) = \frac{\exp(u_i)}{(1+\exp(u_i)}\ ,$$

$$f(v_i) = \frac{1}{|\sigma|}\,\frac{\exp(u_i)}{(1+\exp(u_i))^2}\ ;$$

- Gaussian, $$F(v_i) = \Phi(u_i)$$

(Φ is the standard Normal distribution function),

$$f(v_i) = \frac{1}{|\sigma|\sqrt{2\pi}}\exp(-\tfrac{1}{2}\,u_i^2);$$

- extreme value, $$F(v_i) = 1 - \exp(-\exp(u_i)),$$

$$f(v_i) = \frac{1}{|\sigma|}\exp(u_i - \exp(u_i)).$$

In principle, it is possible for σ to be negative, but this contradicts the physical notion of a stress variable.

In the method of maximum likelihood, parameters θ are estimated by maximizing the log likelihood,

$$L(\theta) = \sum_{i=1}^{n} [a_i \log F(v_i) + (1-a_i)\ \log S(v_i)],$$

where $S = 1-F$. The self-critical procedure depends on a user-specified quantity c, and estimates θ by solving the system

$$\sum_{i=1}^{n} f^c(v_i)\left[a_i \frac{\partial}{\partial\theta}\log F(v_i) + (1-a_i)\frac{\partial}{\partial\theta}\log S(v_i)\right] = 0.$$

When $c = 0$, maximum likelihood estimates are obtained. As c increases from zero, increasingly robust estimators result. The sensitivity of parameter estimates to departures from model assumptions can be examined by starting with $c = 0$ and refitting the model for increasing values of c. Negative values of c seem less useful.

4. Numerical Method

For the purpose of estimation, the routine (internally) reparameterizes the problem in a "regression" form. In this parameterization, $F(\cdot)$ depends on

$$w_i = \alpha v_i + \mu' X_i$$

instead of

$$u_i = \frac{v_i - \beta' X_i}{\sigma} .$$

The reparameterization offers two advantages:

1) Computation of the necessary partial derivatives is simple;
2) For maximum likehood estimation with a logistic tolerance distribution, it can be shown that the regression parameterization results in a concave maximization problem. We anticipate that it will have fairly good properties in the more complicated cases.

It has a disadvantage in terms of potential ill-conditioning of the Hessian (or Jacobian) matrix, so that the input data should be sensibly scaled (see Section 11). Parameters are assumed to be expressed in the easier to understand location-scale form on entry, and are transformed back to location-scale form on exit, so the user need not worry about details of reparameterization.

Let

$$S_{\theta i} = f^{c}(v_i)\left[a_i \frac{\partial}{\partial\theta} \log F(v_i) + (1-a_i) \frac{\partial}{\partial\theta} \log S(v_i)\right] .$$

The gradient used in Newton-Raphson iteration has θ^{th} component $g_\theta = n^{-1}\sum_{i=1}^{n} S_{\theta i}$, and the Hessian has (θ, θ') component $H_{\theta\theta'} = n^{-1}\sum_{i=1}^{n} \frac{\partial}{\partial\theta'} S_{\theta i}$, (the scaling by n^{-1} should be helpful when n is large).

When c = 0, the estiamted asymptotic covariance matrix of the estimators is $n^{-1}(-H)^{-1}$, while when $c \neq 0$ it is $n^{-1}H^{-1}VH^{-1}$, where V has (θ,θ') component $V_{\theta\theta'} = n^{-1} \sum_{i=1}^{n} S_{\theta i}S_{\theta' i}$. The estimated asymptotic covariance matrix is expressed in location-scale form by pre- and post-multiplying it by the Jacobian of the underlying parameter transformation.

Newton-Raphson iteration converges when

$$\max_{i} \left| \theta^{(k)} - \theta_{i}^{(k-1)} \right| < \text{ABSTOL}$$

and

$$\max_{i} \left| \frac{\theta_{i}^{(k)}}{\theta^{(k-1)}} \right| < \text{RELTOL},$$

where superscripts represent iteration numbers and ABSTOL and RELTOL are user-supplied. The user also specifies a maximum allowable number MAXIT of iterations.

Solution of linear equations and matrix inversion is accomplished by subroutines DECOMP and SOLVE, taken from Forsythe, Malcolm and Moler (1977). These routines are of high numerical quality, and provide an estimate of the condition number of the input matrix, which is often useful. In the interest of portability, iterative improvement is not used.

5. Parameters

5.1 Input Parameters

N - INTEGER
Sample size. Unchanged on exit.

IX - INTEGER
Row dimension of data matrix X. Unchanged on exit.

X - REAL array of DIMENSION (IX, q), where q ≥ N. Data matrix containing the dependent (stress) variable and all covariates. Each column corresponds to one observation. If parameters are held fixed, values of the dependent variable will be changed, but their input values restored.
Unchanged on exit.

IA - INTEGER array of DIMENSION N.
Withstand or failure is indicated for observation I according as IA(I) = 0 or IA(I) ≠ 0.
Unchanged on exit.

NPAR - INTEGER.
Number of parameters in the model.
Unchanged on exit.

ISUB - INTEGER array of DIMENSION (NPAR).
Indicates rows of data matrix X corresponding to parameters (the scale parameter is taken to correspond to the dependent variable). For instance, if ISUB(3) = 5, the third parameter corresponds to the fifth row of X.
Unchanged on exit.

ISTART - INTEGER array of DIMENSION (NPAR).
Indicates the status of parameters in the model, and whether a starting value is to be supplied.
If ISTART(I) is equal to
0, BETA(I) is to be estimated, and its input value is to be disregarded;
1, BETA(I) is to be estimated, and its input value is to be used as starting value;
2, BETA(I) is to be held constant at its input value.
(See Section 11.2 for comments on starting values; the scale of parameter is not allowed to be held constant.)
Unchanged on exit.

IDEP - INTEGER.
Row of dependent (stress) variable in X.
Unchanged on exit.

IDIST - INTEGER.
Indicates the tolerance distribution desired.
If IDIST equals
1, logistic distribution will be used;
2, Gaussian distribution will be used;
3, extreme value distribution will be used.
Unchanged on exit.

C - DOUBLE PRECISION
User-supplied constant for self-critical estimation.
Unchanged on exit.

RELTOL - DOUBLE PRECISION.
Relative convergence tolerance for Newton-Raphson iteration (see Section 4).
Unchanged on exit.

MAXIT - INTEGER.
Maximum allowable number of Newton-Raphson iteration.
Unchanged on exit.

IPRINT - INTEGER.
Output unit number. If IPRINT $\leq$ 0, no output will be produced. If IPRINT > 0, standard output will be produced on logical output unit IPRINT.
Unchanged on exit.

IFLAG - INTEGER.
Indicator for form of output.
If IFLAG < 0, the standard output summary will be in regression form. If IFLAG = 0, output summaries will be produced for both regression and location-scale parameterizations. If IFLAG > 0, the standard output summary will be in location-scale form. (See Section 11.3).
Unchanged on exit.

5.2 Input/Output (and associated dimension) Parameters.

BETA - DOUBLE PRECISION array of DIMENSION (NPAR).
On entry, contains starting values as specified by ISTART.
On exit, contains parameter estimates, in location scale form.

XLOGL - DOUBLE PRECISION.
On exit, contains the log likelihood if c = 0 (maximum likelihood estimation), and zero otherwise.

ICOV - INTEGFR.
Row dimenision for COV.
Unchanged on exit.

COV - DOUBLE PRECISION array of DIMENSION (ICOV, q), where q $\geq$ NPAR.
On exit, estimated asymptotic covariance information for the parameters. The diagnal contains standard errors, the strict lower triangle correlations, and the strict upper triangle covariances. If a parameter is held constant, all corresponding entries are zero. The covariance matrix will be for the location-scale parameterization rulers IFLAG < 0, when it will be in regression form. See Section 11.3.

5.3 Workspace (and associated dimension) parameters

LMEM - INTEGER.
Length of work array MEMORY, as declared in calling program unit.
LMEM ≥ NPAR + 4*NACT*(1+NACT) + 2*max(NFIX, 2*NACT), where
NACT is the number of parameters estimated and NFIX = NPAR-NACT
is the number of parameters held constant.
Unchanged on exit.

MEMORY - INTEGER array of DIMENSION (LMEM).
Used as workspace.

5.4 Diagnostic Parameter

IFAULT - INTEGER.
Unless the routine finds an error or gives a warning, IFAULT
will be 0 on exit. See Section 6.

6. Error Indications and Warnings

Errors or warnings specified by the routine:

IFAULT < 0 If IFAULT = -I, I > 0, then an arithmetic exception was about to occur on observation I, while computing partial derivatives. This failure should be rare. If the data have been sensibly scaled, and the starting values are not too bad (see Sections 11.1-11.2), the probable cause is a data error. The routine stops as soon as the error is detected, and output parameter values are not of interest.

IFAULT = 1 Input parameter outside expected range. This failure will occur if, on entry, N < 1, NPAR < 1, IX < NPAR, ICOV < NPAR, RELTOL ≤ 0, ABSTOL < 0, MAXIT < 1, IDIST < 1, IDIST > 3, IFLAG ≤ 0 and IPRINT ≤ 0 (see Section 11.3), ISTART(I) < 0 or ISTART(I) > 2 for some I, ISTART(I) = 2 for all I, ISUB(I) ≠ IDEP for all I, or ISUB(I) = IDEP and ISTART(I) = 2 for some I. (The last restriction means that the scale parameter cannot be held constant.) The routine stops without doing any calculations.

IFAULT = 2 Insufficient workspace. This failure will occur if, on entry, LMEM < NPAR + 4*NACT*(1+NACT) + 2*MAX(NFIX,2*NACT), where NACT is the number of parameters estimated (ISTART(I) < 2), and NFIX = NPAR - NACT is the number of parameters held constant (ISTART(I) = 2). The routine stops without doing any calculations.

IFAULT = 3 The Hessian matrix has become numerically singular during Newton-Raphson iteration. The routine transforms estimates to location-scale form, restores values of the dependent variable if parameters were held fixed, and stops. Output parameter values are not generally of interest.

IFAULT = 4 The Newton-Raphson iteration did not converge to within RELTOL and ABSTOL in the specified MAXIT iterations. The routine transforms estimates to location-scale form, restores values of the dependent variable if parameters were held fixed, and stops. Output parameter values are not generally of interest.

7. Auxiliary Routines

SUBROUTINE PREPAR(IX, N, X, NPAR, BETA, ISUB, ISTART, NACT, IACT, IMAP, PAR, NFIX, IFIXED, IDEP, WORK)

Prepares for iteration by setting up some indexing and working arrays, subtracting the effects of fixed parameters from the dependent variable, and setting initial active parameter values in regression form.

SUBROUTINE NEWTON(C, MAXIT, RELTOL, ABSTOL, DIFFER, XLOGL, NACT, IACT, PAR, IPIVOT, HESS, HESFAC, N, IX, X, IA, IPRINT, LWORK, WORK, LFAULT)

Carries out the Newton-Raphson iteration.

SUBROUTINE VARIAB(C, XLOGL, REGRES, DIFFER, VROUT, NACT, IACT, IMAP, PAR, IPIVOT, HESS, HESFAC, ICOV, NPAR, COV, DET, N, IX, X, IA, WORK)

Computes the estimated asymptotic covariance matrix, transforms it to location-scale form if requested, and computes correlations and standard errors.

SUBROUTINE EXPOST(REGRES, IDEP, NPAR, BETA, ISTART, NACT, PAR, IMAP, N, IX, X, NFIX, IFIXED, WORK)

Sets the output vector BETA, transforms to location-scale form if requested. If location-scale form is requeted and parameters were held fixed, restore initial values of dependent variable.

SUBROUTINE RESULT (C, REGRES, IPRINT, IDIST, N, IDEP, NACT, NFIX, XLOGL, NPAR, BETA, ISUB, ISTART, ICOV, COV, DET)

Produces a standard output summary on logical output unit IPRINT.

The following routines are specific to particular tolerance distributions, and are declared EXTERNAL in BINARY. The first three, prefixed D, are passed to NEWTON and VARIAB. The next three, prefixed V, are passed to VARIAB.

SUBROUTINE DLOGBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX, X, IA,, IFAULT)
SUBROUTINE DGAUBN (——— " ———)
SUBROUTINE DEXVBN (——— " ———)

Compute gradient and Hessian for logistic, Gaussian and extreme value models, resp.

SUBROUTINE VLOGBN(C, NACT, IACT, PAR, V, N, IX, X, IA)
SUBROUTINE VGAUBN(——— " ———)
SUBROUTINE VEXUBN(——— " ———)

Compute V factor of estimated asymptotic covariance matrix for logistic, Guassian and extreme value models, resp.

The following procedures perform general numerical tasks. They have been included in the interest of portability, although equivalents exist on many computer systems.

SUBROUTINE DECOMP(NDIM, N, A, UL, COND, IPUT, WORK)
SUBROUTINE SOLVE(NDIM, N, A, B, X, IPUT)

DECOMP decomposes a matrix into LU factors and estimates its condition. SOLVE solves a linear system, using the results of DECOMP. These routines are in Forsythe, Malcolm and Moler (1977), but the present versions include an extra argument so that A and B need not be overwritten.

SUBROUTINE DMXMLT(A, IA, N1, B, IB, N2, C, IC, N3, WORK, LWORK, IFLAG, IFAULT)

Double precision matrix multiplication -
A(N1 x N3) = B(N1 x N2) * C(N2 x N3), where B is overwritten if IFLAG < 0 and c is overwritten if IFLAG > 0.

DOUBLE PRECISION FUNCTION ALNORM (X, UPPER)
Algorithm as 66 (Hill, 1973) to compute tail areas of the standard Normal curve.

DOUBLE PRECISION FUNCTION RMILLS(X)
Computes Z(x)/Q(x), the reciprocal of Mills' ratio, where x is a standard Normal variate. This procedure is based on procedures by Hill (1973) and Adams (1969).

8. References

Adams, A.G. (1969). Algorithm 39. Areas under the normal curve. Computer J., 12, 197-198.

Forsythe, G.E. Malcolm, M.A. and Moler, C.B. (1977). Computer Methods for Mathematical Computations. Englewood Cliffs, New Jersey: Prentice-Hall.

Hill, I.D. (1973). Algorithm AS 66. The normal integral. Applied Statistics, 22, 424-427.

Paulson, A.S., Presser, M.A. and Lawrence, C.E. (1983). Self-critical, and robust, procedures for the analysis of binary data. Operations Research and Statistics Report No. 37-83-P1, Department of Operations Research and Statistics, Rensselaer Polytechnic Institute.

9. Storage

There are no internally declared arrays.

10. Precision, Machine Dependent Constants

The routein was developed on an IBM computers. The data matrix is single precision to conserve storage. To convert to single precision, take the following steps, with the exception noted below:

1) Change all DOUBLE PRECISION declarations to REAL;
2) Replace references to double precision FORTRAN library functions with single precision versions, e.g., EXP replaces DEXP;
3) Replace double precision constants by their single precision versions, e.g., 1.0 replaces 1.0D0.

Note: In routines PREPAR and EXPOST, the dependent variable is adjusted for the effect of fixed parameters. These adjustements must be calculated in double precision

to avoid a loss of significant digits in X which could affect subsequent computations.

Procedures DLOGBN, VLOGBN, DGAUBN, VGAUBN, DEXUBN, VEXUBN, DECOMP, ALNORM and RMILLS use machine-dependent constants whose vlaues are set in DATA statements. These constants may have to be altered for some computers. They are pointed out by comments in the program units.

11. Further Comments

11.1 Scaling, conditioning of Hessian matrix

The user should be aware of potential problems of ill-conditioning. Because a regression-type parameterization is used, calculation of the Hessian matrix involves operations similar to the formation of $X^T X$, where X is the data matrix. Unfortunately, the resulting Hessian is often rather ill-conditioned. If IPRINT > 0, an estimate of the Hessian's condition number is printed out at each iteration. Roughly speaking, a condition number in excess of 10^7 is worrisome, although condition numbers in excess of 10^{10} can be tolerated when working in double precision on an IBM 3081. The user can and should avoid potentially excessive ill-conditioning by scaling the data matrix X before calling the routine.

Although it is not known how to optimally scale a problem, it seems that a nearly ideal scaling will be achieved if nonconstant variables are transformed to the range [-1, 1], centered at zero. However, such precise scaling if often tedious. It seems most important to "equilibrate" the data matrix so that all variables have roughly the same (moderate) magnitude. Such equilibration is often simply accomplished by dividing by suitable powers of 10, e.g., expressing voltages in megavolts instead of kilovolts. Centering the variables will further reduce the condition number. One often centers covariates anyway, so that the "intercept"

parameter has a clear interpretation.

11.2. Starting values

Since Newton-Raphson procedure is employed, starting values are important. If no starting values are supplied by the user (ISTART(I) = 0 for all I), the routine will use unity for the scale parameter and zero for all location parameters. These starting values will not be acceptable unless the dependent variable has been centered and scaled. It is recommended that the user employ the mean and standard deviation of the dependent variable as starting values for the "intercept" parameter (if any) and scale parameter, respectively. If the model is sequentially refit with different values of c, it is recommended that estimates from the most recent call be used as starting values for the next call.

11.3 Output Flags

Because the routine will generally be called sequentially with different values of c, it is most convenient to always express entry and exit parameter values in the same form. The location-scale form is used, because that form affords the clearest interpretation. An inconsistency arises of IFLAG < 0, for them the output estimates are in location-scale form, but the convariance matrix is in regression form. Thus, if the user wants to make separate use of the output parameters of BINARY, the routine should be called with IFLAG ≥ 0.

It is recommended that the routine be called with IPRINT > 0. If IFLAG ≤ 0, the requirement IPRINT > 0 is enforced. (When IFLAG < 0, the requirement is in keeping with the inconsistency mentioned above.) The standard output summary should

be sufficient for most applications. For the maximum flexibility in output and interpretation of results, call BINARY with IPRINT > 0, IFLAG = 0.

12. Example

The following program illustrates the use of BINARY, and the standard output produced when IPRING > 0 and IFLAG = 0.

```
C*********************************************************************
C       (OPTIONALLY ROBUST) BINARY DATA ESTIMATION ROUTINE.
C       LOGISTIC, EXTREME VALUE, OR GAUSSIAN TOLERANCE DISTRIBUTION
C       CAN BE USED.  PARAMETERS ON INPUT/OUTPUT ARE IN LOCATION-
C       SCALE FORM, BUT ESTIMATION IS CARRIED OUT USING A
C       REGRESSION-TYPE REPARAMETERIZATION.  IF PRINTOUT IS DESIRED
C       (IPRINT .GT. O), RESULTS CAN BE PRINTED OUT IN REGRESSION
C       FORM, LOCATION-SCALE FORM, OR BOTH.  ARGUMENT IFLAG
C       SPECIFIES THE OUTPUT DESIRED.
C       FAILURE CODES -
C               IFAULT = 1 - INPUT ERROR
C               IFAULT = 2 - INSUFFICIENT WORKSPACE SUPPLIED
C               IFAULT = 3 - HESSIAN MATRIX IS NUMERICALLY SINGULAR
C               IFAULT = 4 - NO CONVERGENCE IN MAXIT ITERATIONS
C               IFAULT = -I - AN EXCEPTION WAS ABOUT TO OCCUR WHILE
C                             PROCESSING THE I TH OBSERVATION
C*********************************************************************
      SUBROUTINE BINARY(N, IX, X, IA, NPAR, ISUB, ISTART, IDEP, IDIST,
     1           C, RELTOL, ABSTOL, MAXIT, IPRINT, IFLAG, BETA, XLOGL,
     2           ICOV, COV, LMEM, MEMORY, IFAULT)
C        EXTERNAL PROCEDURES FOR PARTICULAR TOLERANCE DISTRIBUTIONS
      EXTERNAL DLOGBN, DGAUBN, DEXVBN, VLOGBN, VGAUBN, VEXVBN
C        ARGUMENTS
      INTEGER N, IX, IA(N), NPAR, ISUB(NPAR), ISTART(NPAR), IDEP, IDIST,
     1        MAXIT, IPRINT, IFLAG, ICOV, LMEM, MEMORY(LMEM), IFAULT
      REAL X(IX,N)
      DOUBLE PRECISION C, RELTOL, ABSTOL, BETA(NPAR), XLOGL,
     1        COV(ICOV,NPAR)
C        LOCAL SCALARS
      LOGICAL REGRES
      INTEGER NACT, NFIX, IND, IND1, MIACT, MIFIX, MIMAP, MIPVT, MPAR,
     1        MHESS, MHFAC, MWORK
      DOUBLE PRECISION DET, ZERO
C        CONSTANTS
      DATA ZERO /O.ODO/
C
C        CHECK FOR INPUT ERRORS
      IFAULT = 1
      IF (N .LT. 1 .OR. NPAR .LT. 1 .OR. IX .LT. NPAR .OR. ICOV .LT.
     1        NPAR) RETURN
      IF (RELTOL .LE. ZERO .OR. ABSTOL .LE. ZERO .OR. MAXIT .LT. 1)
     1        RETURN
      IF (IDIST .LT. 1 .OR. IDIST .GT. 3) RETURN
      IF (IFLAG .LE. O .AND. IPRINT .LE. O) RETURN
C
C        MEMORY MANAGEMENT, POSSIBLE RELATED INPUT ERRORS.  CHECK
C        ISUB(*), ISTART(*) VECTORS, COUNT ACTIVE PARAMETERS.
C        NOTE SCALE PARAMETER CANNOT BE FIXED.
      NACT = O
      IND1 = O
      DO 10 I = 1, NPAR
        IND = ISUB(I)
        IF (IND .LT. 1 .OR. IND .GT. IX) RETURN
        IF (IND .EQ. IDEP) IND1 = I
        IND = ISTART(I)
        IF (IND .LT. O .OR. IND .GT. 2) RETURN
        IF (IND .NE. 2) NACT = NACT + 1
   10 CONTINUE
      IF (NACT .EQ. O .OR. IND1 .EQ. O) RETURN
      IF (ISTART(IND1) .EQ. 2) RETURN
```

```
      NFIX = NPAR - NACT                                                BINA0061
C                                                                       BINA0062
C        CHECK IF WORKSPACE SIZE IS ADEQUATE, ALLOCATE IT               BINA0063
      IFAULT = 2                                                        BINA0064
      IND = 2 * NACT                                                    BINA0065
      IND1 = IND * NACT                                                 BINA0066
      IF (LMEM .LT. NPAR + 2*NACT + IND + 2*IND1 + MAX0(2*NFIX,2*IND))  BINA0067
     1      RETURN                                                      BINA0068
      MIACT = 1                                                         BINA0069
      MIFIX = MIACT + NACT                                              BINA0070
      MIMAP = MIACT + NPAR                                              BINA0071
      MIPVT = MIMAP + NACT                                              BINA0072
      MPAR = MIPVT + NACT                                               BINA0073
      MHESS = MPAR + IND                                                BINA0074
      MHFAC = MHESS + IND1                                              BINA0075
      MWORK = MHFAC + IND1                                              BINA0076
C                                                                       BINA0077
C        PREPARE FOR ESTIMATION - SET UP SUBSCRIPT ARRAYS AND           BINA0078
C        STARTING VALUES FOR ACTIVE PARAMETERS, SUBTRACT EFFECT OF      BINA0079
C        FIXED PARAMETERS FROM DEPENDENT VARIABLE                       BINA0080
      CALL PREPAR(IX, N, X, NPAR, BETA, ISUB, ISTART, NACT,             BINA0081
     1    MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR), NFIX,             BINA0082
     2    MEMORY(MIFIX), IDEP, MEMORY(MWORK))                           BINA0083
C                                                                       BINA0084
C        NEWTON-RAPHSON ITERATION WITH EXTERNAL ROUTINE PASSED FOR      BINA0085
C        FIRST AND SECOND PARTIALS                                      BINA0086
      IF (IDIST .EQ. 1) CALL NEWTON(C, MAXIT, RELTOL, ABSTOL, DLOGBN,   BINA0087
     1    XLOGL, NACT, MEMORY(MIACT), MEMORY(MPAR), MEMORY(MIPVT),      BINA0088
     2    MEMORY(MHESS), MEMORY(MHFAC), N, IX, X, IA, IPRINT, 2*NACT,   BINA0089
     3    MEMORY(MWORK), IFAULT)                                        BINA0090
      IF (IDIST .EQ. 2) CALL NEWTON(C, MAXIT, RELTOL, ABSTOL, DGAUBN,   BINA0091
     1    XLOGL, NACT, MEMORY(MIACT), MEMORY(MPAR), MEMORY(MIPVT),      BINA0092
     2    MEMORY(MHESS), MEMORY(MHFAC), N, IX, X, IA, IPRINT, 2*NACT,   BINA0093
     3    MEMORY(MWORK), IFAULT)                                        BINA0094
      IF (IDIST .EQ. 3) CALL NEWTON(C, MAXIT, RELTOL, ABSTOL, DEXVBN,   BINA0095
     1    XLOGL, NACT, MEMORY(MIACT), MEMORY(MPAR), MEMORY(MIPVT),      BINA0096
     2    MEMORY(MHESS), MEMORY(MHFAC), N, IX, X, IA, IPRINT, 2*NACT,   BINA0097
     3    MEMORY(MWORK), IFAULT)                                        BINA0098
C                                                                       BINA0099
C        ERROR HANDLING IF NEWTON-RAPHSON PROCEDURE FAILS.              BINA0100
C        ON ERROR, PRINT OUT MESSAGE IF IPRINT .GT. 0, SET UP           BINA0101
C        THE FINAL CALL TO EXPOST().                                    BINA0102
      IF (IFAULT .EQ. 0) GO TO 30                                       BINA0103
      REGRES = .FALSE.                                                  BINA0104
      IF (IPRINT .LE. 0) GO TO 50                                       BINA0105
      IF (IFAULT .GT. 0) GO TO 20                                       BINA0106
      IND = -IFAULT                                                     BINA0107
      WRITE (IPRINT,60) IND                                             BINA0108
      GO TO 50                                                          BINA0109
   20 IFAULT = IFAULT + 1                                               BINA0110
      IF (IFAULT .EQ. 3) WRITE (IPRINT,70)                              BINA0111
      IF (IFAULT .EQ. 4) WRITE (IPRINT,80) MAXIT                        BINA0112
      GO TO 50                                                          BINA0113
C                                                                       BINA0114
C        COMPUTE APPROXIMATE ASYMPTOTIC COVARIANCE MATRIX.              BINA0115
C        REGRES IS A FLAG FOR WHETHER OR NOT TO SET UP OUTPUT IN        BINA0116
C        REGRESSION FORM.  STATEMENT 30, THE FIRST BRANCH POINT,        BINA0117
C        SETS UP REGRES FOR FIRST OUTPUT (IF REGRESSION AND             BINA0118
C        LOCATION-SCALE BOTH REQUESTED, REGRESSION GOES FIRST).         BINA0119
C        STATEMENT 40, THE SECOND BRANCH POINT, IS FOR REPEAT           BINA0120
```

```
C         CALL (ALWAYS LOCATION-SCALE)                                   BINA0121
   30 REGRES = IFLAG .LE. 0                                              BINA0122
   40 IF (IDIST .EQ. 1) CALL VARIAB(C, XLOGL, REGRES, DLOGBN, VLOGBN,    BINA0123
     1      NACT, MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR),            BINA0124
     2      MEMORY(MIPVT), MEMORY(MHESS), MEMORY(MHFAC), ICOV, NPAR, COV,BINA0125
     3      DET, N, IX, X, IA, MEMORY(MWORK))                            BINA0126
      IF (IDIST .EQ. 2) CALL VARIAB(C, XLOGL, REGRES, DGAUBN, VGAUBN,    BINA0127
     1      NACT, MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR),            BINA0128
     2      MEMORY(MIPVT), MEMORY(MHESS), MEMORY(MHFAC), ICOV, NPAR, COV,BINA0129
     3      DET, N, IX, X, IA, MEMORY(MWORK))                            BINA0130
      IF (IDIST .EQ. 3) CALL VARIAB(C, XLOGL, REGRES, DEXVBN, VEXVBN,    BINA0131
     1      NACT, MEMORY(MIACT), MEMORY(MIMAP), MEMORY(MPAR),            BINA0132
     2      MEMORY(MIPVT), MEMORY(MHESS), MEMORY(MHFAC), ICOV, NPAR, COV,BINA0133
     3      DET, N, IX, X, IA, MEMORY(MWORK))                            BINA0134
C                                                                        BINA0135
C         ARRANGE ESTIMATED PARAMETERS IN BETA(*) FOR OUTPUT.            BINA0136
C         IF LOCATION-SCALE FORM, TRANSFORM PARAMETERS AND POSSIBLY      BINA0137
C         RESTORE DEPENDENT VARIABLE TO VALUE ON ENTRY.                  BINA0138
   50 CALL EXPOST(REGRES, IDEP, NPAR, BETA, ISTART, NACT, MEMORY(MPAR),  BINA0139
     1      MEMORY(MIMAP), N, IX, X, NFIX, MEMORY(MIFIX), MEMORY(MWORK)) BINA0140
C                                                                        BINA0141
C         IF NEWTON-RAPHSON FAILED, GOODBYE.                             BINA0142
      IF (IFAULT .NE. 0) RETURN                                          BINA0143
C                                                                        BINA0144
C         PRODUCE STANDARD OUTPUT IF REQUESTED.                          BINA0145
      IF (IPRINT .GT. 0) CALL RESULT(C, REGRES, IPRINT, IDIST, N, IDEP,  BINA0146
     1      NACT, NFIX, XLOGL, NPAR, BETA, ISUB, ISTART, ICOV, COV, DET) BINA0147
C                                                                        BINA0148
C         IF OUTPUT DESIRED IN BOTH REGRESSION AND LOCATION-SCALE        BINA0149
C         FORMS, LOOP BACK, ELSE EXIT                                    BINA0150
      REGRES = .NOT. REGRES                                              BINA0151
      IF (.NOT. REGRES .AND. IFLAG .EQ. 0) GO TO 40                      BINA0152
C                                                                        BINA0153
C         IF ONLY REGRESSION FORM OUTPUT WAS REQUESTED, AN EXTRA         BINA0154
C         CALL TO EXPOST() IS NEEDED TO RESTORE LOCATION-SCALE FORM      BINA0155
      IF (IFLAG .LT. 0) CALL EXPOST(REGRES, IDEP, NPAR, BETA, ISTART,    BINA0156
     1      NACT, MEMORY(MPAR), MEMORY(MIMAP), N, IX, X, NFIX,           BINA0157
     2      MEMORY(MIFIX), MEMORY(MWORK))                                BINA0158
C                                                                        BINA0159
   60 FORMAT ('0FAILURE - EXCEPTION ON OBSERVATION NO.', I5)             BINA0160
   70 FORMAT ('0FAILURE - HESSIAN MATRIX IS NUMERICALLY SINGULAR')       BINA0161
   80 FORMAT ('0WARNING - NO CONVERGENCE IN', I4, ' ITERATIONS - POSSIBLBINA0162
     1E FAILURE')                                                        BINA0163
      RETURN                                                             BINA0164
      END                                                                BINA0165
C***********************************************************************PREP0001
C         PREPARE FOR ANALYSIS BY SETTING UP THE FOLLOWING ARRAYS -      PREP0002
C             IACT(*) - X(*,*) ROWS FOR ACTIVE PARAMETERS                PREP0003
C             IFIXED(*) - X(*,*) ROWS FOR FIXED PARAMETERS               PREP0004
C             IMAP(*) - POSITIONS IN BETA(*) OF ACTIVE PARAMETERS        PREP0005
C             PAR(*) - INITIAL VALUES OF ACTIVE PARAMETERS IN            PREP0006
C                       REGRESSION FORM                                  PREP0007
C         ALSO SUBTRACT EFFECTS OF FIXED PARAMETERS FROM DEPENDENT       PREP0008
C         VARIABLE, USING WORK(*)                                        PREP0009
C         NOTE - DEPENDENT VARIABLE (SCALE) PARAMETER IS ALWAYS          PREP0010
C                PLACED FIRST IN PAR(*), FOR CONVENIENCE LATER           PREP0011
C***********************************************************************PREP0012
      SUBROUTINE PREPAR(IX, N, X, NPAR, BETA, ISUB, ISTART, NACT, IACT,  PREP0013
     1          IMAP, PAR, NFIX, IFIXED, IDEP, WORK)                     PREP0014
C         ARGUMENTS                                                      PREP0015
```

```
      INTEGER IX, N, NPAR, ISUB(NPAR), ISTART(NPAR), NACT, IACT(NACT),  PREP0016
     1        IMAP(NACT), NFIX, IFIXED(1), IDEP                         PREP0017
      REAL X(IX,N)                                                      PREP0018
      DOUBLE PRECISION BETA(NPAR), PAR(NACT), WORK(1)                   PREP0019
C        LOCAL SCALARS                                                  PREP0020
      INTEGER IND, IND1, LACT, ISCALE                                   PREP0021
      DOUBLE PRECISION TEMP, ZERO, ONE                                  PREP0022
      DATA ZERO, ONE /0.0D0, 1.0D0/                                     PREP0023
C                                                                       PREP0024
C        SET UP IACT(*), IFIXED(*), IMAP(*)                             PREP0025
      LACT = 0                                                          PREP0026
      IND = 0                                                           PREP0027
      ISCALE = 0                                                        PREP0028
      DO 20 I = 1, NPAR                                                 PREP0029
        IF (ISTART(I) .EQ. 2) GO TO 10                                  PREP0030
        LACT = LACT + 1                                                 PREP0031
        IMAP(LACT) = I                                                  PREP0032
        IND1 = ISUB(I)                                                  PREP0033
        IACT(LACT) = IND1                                               PREP0034
        IF (IND1 .EQ. IDEP) ISCALE = I                                  PREP0035
        GO TO 20                                                        PREP0036
   10   IND = IND + 1                                                   PREP0037
        IFIXED(IND) = ISUB(I)                                           PREP0038
        WORK(IND) = BETA(I)                                             PREP0039
   20 CONTINUE                                                          PREP0040
C                                                                       PREP0041
C        SWITCH PLACES SO SCALE IS FIRST ACTIVE PARAMETER               PREP0042
      IND = IACT(1)                                                     PREP0043
      IACT(1) = IACT(ISCALE)                                            PREP0044
      IACT(ISCALE) = IND                                                PREP0045
      IND = IMAP(1)                                                     PREP0046
      IMAP(1) = IMAP(ISCALE)                                            PREP0047
      IMAP(ISCALE) = IND                                                PREP0048
C                                                                       PREP0049
C        MAIN LOOP OVER SAMPLE IF PARAMETERS ARE FIXED, TO ADJUST       PREP0050
C        DEPENDENT VARIABLE (TO BE RESET ON EXIT FROM BINARY())         PREP0051
      IF (NFIX .EQ. 0) GO TO 50                                         PREP0052
      DO 40 I = 1, N                                                    PREP0053
C                                                                       PREP0054
C        THE FOLLOWING INNER PRODUCT MUST BE ACCUMULATED IN DOUBLE      PREP0055
C        PRECISION...                                                   PREP0056
        TEMP = DBLE(X(IDEP,I))                                          PREP0057
        DO 30 J = 1, NFIX                                               PREP0058
          IND = IFIXED(J)                                               PREP0059
          TEMP = TEMP - WORK(J) * X(IND,I)                              PREP0060
   30   CONTINUE                                                        PREP0061
        X(IDEP,I) = SNGL(TEMP)                                          PREP0062
C                                                                       PREP0063
   40 CONTINUE                                                          PREP0064
C                                                                       PREP0065
C        SET STARTING VALUES                                            PREP0066
   50 DO 60 I = 1, NPAR                                                 PREP0067
   60 PAR(I) = ZERO                                                     PREP0068
      TEMP = ONE                                                        PREP0069
      IND = IMAP(1)                                                     PREP0070
      IF (ISTART(IND) .EQ. 1 .AND. BETA(IND) .NE. ZERO) TEMP = BETA(IND)PREP0071
      PAR(1) = ONE / TEMP                                               PREP0072
      IF (NACT .EQ. 1) RETURN                                           PREP0073
      DO 70 I = 2, NACT                                                 PREP0074
        IND = IMAP(I)                                                   PREP0075
```

```
      IF (ISTART(IND) .EQ. 1) PAR(I) = -BETA(IND) / TEMP
   70 CONTINUE
C
      RETURN
      END
C**********************************************************************
C        NEWTON-RAPHSON ITERATION
C        FAILURE CODES -
C        IFAULT = 1 - INSUFFICIENT WORKSPACE (LESS THAN 2 * NACT
C                     LOCATIONS)
C        IFAULT = 2 - THE HESSIAN MATRIX IS NUMERICALLY SINGULAR
C        IFAULT = 3 - CONVERGENCE HAS NOT OCCURRED IN MAXIT ITNS
C        IFAULT = -I - AN EXCEPTION WAS ABOUT TO OCCUR WHILE
C                      PROCESSING THE I TH OBSERVATION IN DIFFER()
C**********************************************************************
      SUBROUTINE NEWTON(C, MAXIT, RELTOL, ABSTOL, DIFFER, XLOGL, NACT,
     1           IACT, PAR, IPIVOT, HESS, HESFAC, N, IX, X, IA, IPRINT,
     2           LWORK, WORK, IFAULT)
C        ARGUMENTS - DIFFER IS EXTERNAL ROUTINE FOR PARTIALS
      INTEGER MAXIT, NACT, IACT(NACT), IPIVOT(NACT), N, IX, IA(N),
     1        IPRINT, LWORK, IFAULT
      REAL X(IX,N)
      DOUBLE PRECISION C, RELTOL, ABSTOL, XLOGL, PAR(NACT),
     1       HESS(NACT,NACT), HESFAC(NACT,NACT), WORK(LWORK)
C        LOCAL SCALARS
      INTEGER ITER, IND
      DOUBLE PRECISION GNORM, COND, ABSERR, RELERR, TEMP, XNEW, ZERO,
     1       ONE
      DATA ZERO, ONE /0.0D0, 1.0D0/
C
      IFAULT = 1
      IF (LWORK .LT. 2*NACT) RETURN
      IFAULT = 0
      IND = NACT + 1
      ITER = 0
      IF (IPRINT .GT. 0) WRITE (IPRINT,50) C, MAXIT, RELTOL, ABSTOL
C
C        LOOPING POINT FOR ITERATION . THE FIRST NACT LOCATIONS OF
C        WORK(*) ARE USED FOR GRADIENT/INCREMENT, WHILE THE NEXT
C        NACT LOCATIONS ARE WORKSPACE FOR DECOMP
   10 ITER = ITER + 1
      CALL DIFFER(C, XLOGL, NACT, IACT, PAR, WORK(1), HESS, N, IX, X,
     1      IA, IFAULT)
      IF (IFAULT .LT. 0) RETURN
C
C        FACTOR THE HESSIAN INTO L * U, CHECK IF SINGULAR
      CALL DECOMP(NACT, NACT, HESS, HESFAC, COND, IPIVOT, WORK(IND))
      IF (COND + ONE .NE. COND) GO TO 20
      IFAULT = 2
      RETURN
C
C        SOLVE THE SYSTEM HESS * INCREMENT = -GRADIENT,
C        OVERWRITING GRADIENT
   20 GNORM = ZERO
      DO 30 I = 1, NACT
        TEMP = WORK(I)
        WORK(I) = -TEMP
        GNORM = GNORM + TEMP * TEMP
   30 CONTINUE
      GNORM = DSQRT(GNORM)
```

```
      CALL SOLVE(NACT, NACT, HESFAC, WORK, WORK, IPIVOT)
C
C         INCREMENT PARAMETERS, COMPUTE CONVERGENCE CRITERIA
      RELERR = ZERO
      ABSERR = ZERO
      DO 40 I = 1, NACT
        TEMP = WORK(I)
        ABSERR = DMAX1(ABSERR,DABS(TEMP))
        XNEW = PAR(I) + TEMP
        PAR(I) = XNEW
        IF (ONE + XNEW .NE. ONE) TEMP = TEMP / XNEW
        RELERR = DMAX1(RELERR,DABS(TEMP))
   40 CONTINUE
      IF (IPRINT .GT. 0) WRITE (IPRINT,60) ITER, GNORM, RELERR, ABSERR,
     1COND
C
C         CHECK CONVERGENCE, SET FAILURE CODE IF NONE
      IF (RELERR .LT. RELTOL .AND. ABSERR .LT. ABSTOL) RETURN
      IF (ITER .LT. MAXIT) GO TO 10
      IFAULT = 3
C
   50 FORMAT ('1   C        MAX ITNS  REL TOLER  ABS TOLER'/D11.3, I9,
     1        2D11.2)
   60 FORMAT ('0 ITN   GRAD NORM  REL CHANGE  ABS CHANGE'/I5, 3D12.2, /,
     1        ' ESTIMATED CONDITION NUMBER OF HESSIAN IS', D10.2)
      RETURN
      END
C***********************************************************************
C         COMPUTATIONS FOR VARIABILITY OF THE ESTIMATORS.
C         ESTIMATE ASYMPTOTIC COVARIANCE MATRIX, TRANSFORM IT FROM
C         REGRESSION TO LOCATION-SCALE FORM, COMPUTE ASYMPTOTIC
C         CORRELATIONS AND STD. ERRORS.
C***********************************************************************
      SUBROUTINE VARIAB(C, XLOGL, REGRES, DIFFER, VROUT, NACT, IACT,
     1           IMAP, PAR, IPIVOT, HESS, HESFAC, ICOV, NPAR, COV, DET,
     2           N, IX, X, IA, WORK)
C         ARGUMENTS - DIFFER AND VROUT ARE SUBROUTINES
      LOGICAL REGRES
      INTEGER NACT, IACT(NACT), IMAP(NACT), IPIVOT(NACT), ICOV, NPAR, N,
     1        IX, IA(N)
      REAL X(IX,N)
      DOUBLE PRECISION C, XLOGL, PAR(NACT), HESS(NACT,NACT),
     1        HESFAC(NACT,NACT), COV(ICOV,NPAR), DET, WORK(NACT)
C         LOCAL SCALARS
      LOGICAL FLAG
      INTEGER IND, IND1, IND2
      DOUBLE PRECISION ALPHA, ALPHA2, TEMP, TEMP1, ZERO, ONE, SMALL
C         MACHINE-DEPENDENT CONSTANT - SMALL SET SO THAT DEXP(X)
C         WILL CAUSE EXCEPTION IF X .LT. SMALL
      DATA ZERO, ONE, SMALL /0.0D0, 1.0D0, -180.0D0/
C
C         CALCULATE HESSIAN AT OPTIMAL POINT AND FACTOR IT
C         (THIS MAY BE WASTEFUL IN SOME CASES, BUT IT AVOIDS
C         SOME LOGICAL COMPLICATIONS.)  WORK(*) USED AS SCRATCH
C         THE HESSIAN IS ASSUMED NONSINGULAR, AS IT SHOULD BE IF
C         THIS POINT IS REACHED.  FACTORIZATION TO HESFAC(*,*)
      CALL DIFFER(C, XLOGL, NACT, IACT, PAR, WORK, HESS, N, IX, X, IA,
     1     IND)
      CALL DECOMP(NACT, NACT, HESS, HESFAC, TEMP, IPIVOT, WORK)
C
```

```
C        INVERT NEGATIVE OF HESSIAN, USING IPIVOT(*) AND                VARI0034
C        FACTORIZATION IN HESFAC(*,*). PLACE INVERSE IN HESS(*,*)       VARI0035
      DO 20 J = 1, NACT                                                 VARI0036
        DO 10 I = 1, NACT                                               VARI0037
   10   HESS(I,J) = ZERO                                                VARI0038
        HESS(J,J) = -ONE                                                VARI0039
        CALL SOLVE(NACT, NACT, HESFAC, HESS(1,J), HESS(1,J), IPIVOT)    VARI0040
   20 CONTINUE                                                          VARI0041
      IF (C  EQ  ZERO) GO TO 30                                         VARI0042
C                                                                       VARI0043
C        SECTION FOR ROBUST ANALYSIS -                                  VARI0044
C        PLACE PRODUCT (H INVERSE) * V * (H INVERSE) IN HESS(*,*)       VARI0045
C        FIRST COMPUTE V(*,*), PLACE IN COV(*,*)                        VARI0046
      CALL VROUT(C, NACT, IACT, PAR, COV, N, IX, X, IA)                 VARI0047
C                                                                       VARI0048
C        MULTIPLY HESS * COV, OVERWRITING COV(*,*)                      VARI0049
      CALL DMXMLT(COV, NACT, NACT, HESS, NACT, NACT, COV, NACT, NACT,   VARI0050
     1     WORK, NACT, 1, IND)                                          VARI0051
C                                                                       VARI0052
C        MULTIPLY COV * HESS, OVERWRITING HESS(*,*)                     VARI0053
      CALL DMXMLT(HESS, NACT, NACT, COV, NACT, NACT, HESS, NACT, NACT,  VARI0054
     1     WORK, NACT, 1, IND)                                          VARI0055
C                                                                       VARI0056
C        ALL VALUES OF C - IF LOCATION-SCALE FORM, TRANSFORM            VARI0057
C        THE COVARIANCE MATRIX.                                         VARI0058
   30 IF (REGRES) GO TO 80                                              VARI0059
      ALPHA = PAR(1)                                                    VARI0060
      ALPHA2 = ALPHA * ALPHA                                            VARI0061
      FLAG = NACT  EQ  1                                                VARI0062
C                                                                       VARI0063
C        LEFT-MULTIPLY HESS(*,*) BY JACOBIAN, RESULT TO HESFAC(*,*)     VARI0064
      DO 50 J = 1, NACT                                                 VARI0065
        TEMP = HESS(1,J) / ALPHA2                                       VARI0066
        HESFAC(1,J) = -TEMP                                             VARI0067
        IF (FLAG) GO TO 50                                              VARI0068
        DO 40 I = 2, NACT                                               VARI0069
   40   HESFAC(I,J) = PAR(I) * TEMP - HESS(I,J) / ALPHA                 VARI0070
   50 CONTINUE                                                          VARI0071
C                                                                       VARI0072
C        RIGHT-MULTIPLY HESFAC(*,*) BY TRANSPOSE OF JACOBIAN,           VARI0073
C        RESULT TO HESS(*,*)                                            VARI0074
      DO 70 I = 1, NACT                                                 VARI0075
        TEMP = HESFAC(I,1) / ALPHA2                                     VARI0076
        HESS(I,1) = -TEMP                                               VARI0077
        IF (FLAG) GO TO 70                                              VARI0078
        DO 60 J = 2, NACT                                               VARI0079
   60   HESS(I,J) = PAR(J) * TEMP   HESFAC(I,J) / ALPHA                 VARI0080
   70 CONTINUE                                                          VARI0081
C                                                                       VARI0082
C        BOTH PARAMETERIZATIONS - DIVIDE COVARIANCE MATRIX BY           VARI0083
C        SAMPLE SIZE                                                    VARI0084
   80 TEMP = DBLE(FLOAT(N))                                             VARI0085
      DO 90 J = 1, NACT                                                 VARI0086
        DO 90 I = 1, J                                                  VARI0087
          HESS(I,J) = HESS(I,J) / TEMP                                  VARI0088
          HESS(J,I) = HESS(I,J)                                         VARI0089
   90 CONTINUE                                                          VARI0090
C                                                                       VARI0091
C        FIND DETERMINANT OF COVARIANCE MATRIX - FACTOR IT,             VARI0092
C        RESETTING IPIVOT(*) AND HESFAC(*,*)                            VARI0093
```

```
      CALL DECOMP(NACT, NACT, HESS, HESFAC, TEMP, IPIVOT, WORK)
      DET = ZERO
      IF (TEMP + ONE .EQ. TEMP) GO TO 110
      IND = IPIVOT(NACT)
      DO 100 I = 1, NACT
        TEMP1 = HESFAC(I,I)
        IF (TEMP1 .LT. ZERO) IND = -IND
        DET = DET + DLOG(DABS(TEMP1))
  100 CONTINUE
C
C        TAKE THE NACT ROOT OF DETERMINANT.  SET DET TO ZERO
C        IF IT UNDERFLOWS OR IS NEGATIVE.
      DET = DET / DBLE(FLOAT(NACT))
      IF (DET .LT. SMALL .OR. IND .LT. 0) DET = ZERO
      IF (DET .GE. SMALL .AND. IND .GT. 0) DET = DEXP(DET)
C
C        MOVE CONTENTS OF HESS(*,*) TO UPPER TRIANGLE OF COV(*,*)
  110 DO 120 J = 1, NPAR
        DO 120 I = 1, NPAR
  120 COV(I,J) = ZERO
      DO 130 I = 1, NACT
        IND = IMAP(I)
        DO 130 J = I, NACT
          IND1 = IMAP(J)
          IND2 = MIN0(IND,IND1)
          IND1 = MAX0(IND,IND1)
          COV(IND2,IND1) = HESS(I,J)
  130 CONTINUE
C
C        PUT STD ERRORS ON DIAG OF COV(*,*), CORRELATIONS BELOW
      DO 150 I = 1, NPAR
        TEMP = COV(I,I)
        IF (TEMP .EQ. ZERO) GO TO 150
        TEMP = DSQRT(TEMP)
        COV(I,I) = TEMP
        IF (I .EQ. 1) GO TO 150
        IND = I - 1
        DO 140 J = 1, IND
          TEMP1 = COV(J,J)
          IF (TEMP1 .NE. ZERO) COV(I,J) = COV(J,I) / (TEMP*TEMP1)
  140   CONTINUE
  150 CONTINUE
C
      RETURN
      END
C****************************************************************
C        EX POST ADJUSTMENTS BEFORE EXIT
C        PUT OPTIMAL PARAMETERS IN BETA(*).  IF LOCATION-SCALE
C        FORM, TRANSFORM THE PARAMETERS AND RESTORE DEPENDENT
C        VARIABLE TO INITIAL VALUES IF PARAMETERS WERE FIXED.
C****************************************************************
      SUBROUTINE EXPOST(REGRES, IDEP, NPAR, BETA, ISTART, NACT, PAR,
     1                  IMAP, N, IX, X, NFIX, IFIXED, WORK)
C        ARGUMENTS
      LOGICAL REGRES
      INTEGER IDEP, NPAR, ISTART(NPAR), NACT, IMAP(NACT), N, IX, NFIX,
     1        IFIXED(1)
      REAL X(IX,N)
      DOUBLE PRECISION BETA(NPAR), PAR(NACT), WORK(1)
C        LOCAL SCALARS
```

```
      INTEGER IND                                                       EXP00016
      DOUBLE PRECISION TEMP, ONE                                        EXP00017
      DATA ONE /1.0D0/                                                  EXP00018
C                                                                       EXP00019
C        INSERT WORKING PARAMETERS PAR(*) IN BETA(*), IN THE            EXP00020
C        SUITABLE FORM.                                                 EXP00021
      TEMP = PAR(1)                                                     EXP00022
      IND = IMAP(1)                                                     EXP00023
      IF (REGRES) BETA(IND) = TEMP                                      EXP00024
      IF (.NOT. REGRES) BETA(IND) = ONE / TEMP                          EXP00025
      IF (NACT .EQ. 1) GO TO 20                                         EXP00026
      DO 10 I = 2, NACT                                                 EXP00027
        IND = IMAP(I)                                                   EXP00028
        IF (REGRES) BETA(IND) = PAR(I)                                  EXP00029
        IF (.NOT. REGRES) BETA(IND) = -PAR(I) / TEMP                    EXP00030
   10 CONTINUE                                                          EXP00031
C                                                                       EXP00032
C        IF LOCATION-SCALE FORM AND PARAMETERS WERE HELD FIXED,         EXP00033
C        RESTORE DEPENDENT VARIABLE                                     EXP00034
   20 IF (REGRES .OR. NFIX .EQ. 0) RETURN                               EXP00035
      IND = 0                                                           EXP00036
      DO 30 I = 1, NPAR                                                 EXP00037
        IF (ISTART(I) .NE. 2) GO TO 30                                  EXP00038
        IND = IND + 1                                                   EXP00039
        WORK(IND) = BETA(I)                                             EXP00040
   30 CONTINUE                                                          EXP00041
      DO 50 I = 1, N                                                    EXP00042
C                                                                       EXP00043
C        THE FOLLOWING INNER PRODUCT MUST BE ACCUMULATED IN DOUBLE      EXP00044
C        PRECISION                                                      EXP00045
        TEMP = DBLE(X(IDEP,I))                                          EXP00046
        DO 40 J = 1, NFIX                                               EXP00047
          IND = IFIXED(J)                                               EXP00048
          TEMP = TEMP + WORK(J) * X(IND,I)                              EXP00049
   40   CONTINUE                                                        EXP00050
        X(IDEP,I) = SNGL(TEMP)                                          EXP00051
C                                                                       EXP00052
   50 CONTINUE                                                          EXP00053
C                                                                       EXP00054
      RETURN                                                            EXP00055
      END                                                               EXP00056
C***********************************************************************RESU0001
C        PRINT OUT TYPICAL OUTPUT SUMMARY ON OUTPUT UNIT IPRINT         RESU0002
C***********************************************************************RESU0003
      SUBROUTINE RESULT(C, REGRES, IPRINT, IDIST, N, IDEP, NACT, NFIX,  RESU0004
     1                  XLOGL, NPAR, BETA, ISUB, ISTART, ICOV, COV, DET) RESU0005
C        ARGUMENTS                                                      RESU0006
      LOGICAL REGRES                                                    RESU0007
      INTEGER IPRINT, IDIST, N, IDEP, NACT, NFIX, NPAR, ISUB(NPAR),     RESU0008
     1           ISTART(NPAR), ICOV                                     RESU0009
      DOUBLE PRECISION C, XLOGL, BETA(NPAR), COV(ICOV,NPAR), DET        RESU0010
C        LOCAL SCALARS                                                  RESU0011
      LOGICAL MLE                                                       RESU0012
      INTEGER IND                                                       RESU0013
      DOUBLE PRECISION TSTAT, ZERO                                      RESU0014
      DATA ZERO /0.0D0/                                                 RESU0015
C                                                                       RESU0016
C        BASIC HEADINGS                                                 RESU0017
      MLE = C .EQ. ZERO                                                 RESU0018
      IF (MLE) WRITE (IPRINT,40)                                        RESU0019
```

```
C       FAILURE CODE - IFAULT = -I - AN EXCEPTION WAS ABOUT TO
C                OCCUR WHILE PROCESSING THE I TH OBSERVATION
C*****************************************************************
      SUBROUTINE DLOGBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX,
     1           X, IA, IFAULT)
C       ARGUMENTS
      INTEGER NACT, IACT(NACT), N, IX, IA(N), IFAULT
      REAL X(IX,N)
      DOUBLE PRECISION C, OBJECT, PAR(NACT), GRAD(NACT), HESS(NACT,NACT)
C       LOCAL SCALARS
      LOGICAL MLE
      INTEGER IND
      DOUBLE PRECISION EXTRA, F, S, G, FC, EITHER, DDENS, XJI, ZERO,
     1       ONE, BIG
C       MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT
C       DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG
      DATA ZERO, ONE, BIG /0.0D0, 1.0D0, 174.0D0/
C
      IFAULT = 0
      MLE = ONE + C .EQ. ONE
      EXTRA = ONE / PAR(1)
      OBJECT = ZERO
      DO 10 J = 1, NACT
        GRAD(J) = ZERO
        DO 10 I = J, NACT
          HESS(I,J) = ZERO
   10 CONTINUE
C
C       MAIN LOOP OVER SAMPLE
      DO 90 I = 1, N
        FC = ZERO
        DO 20 J = 1, NACT
          IND = IACT(J)
          FC = FC + PAR(J) * X(IND,I)
   20   CONTINUE
        IF (DABS(FC) .GT. BIG) GO TO 110
        FC = DEXP(FC)
        S = ONE + FC
        F = FC / S
        S = ONE / S
        G = F * S
        IF (IA(I) .EQ. 0) GO TO 30
        EITHER = S
        XJI = F
        GO TO 40
   30   EITHER = -F
        XJI = S
   40   IF (MLE) GO TO 50
        FC = G ** C
        GO TO 60
   50   OBJECT = OBJECT + DLOG(XJI)
        FC = ONE
C
C        LOOP TO INCREMENT GRADIENT AND HESSIAN UPPER TRIANGLE
   60   EITHER = FC * EITHER
        G = FC * G
        FC = C * EITHER
        DO 80 J = 1, NACT
          IND = IACT(J)
          XJI = DBLE(X(IND,I))
```

```
      IF ( NOT  MLE) WRITE (IPRINT,50) C                                RESU0020
      IF (IDIST  EQ  1) WRITE (IPRINT,60)                               RESU0021
      IF (IDIST  EQ  2) WRITE (IPRINT,70)                               RESU0022
      IF (IDIST  EQ  3) WRITE (IPRINT,80)                               RESU0023
      WRITE (IPRINT,90) N, NPAR, NACT, NFIX                             RESU0024
      IF (REGRES) WRITE (IPRINT,100)                                    RESU0025
      IF ( NOT  REGRES) WRITE (IPRINT,110)                              RESU0026
C                                                                       RESU0027
C         PARAMETER ESTIMATES                                           RESU0028
      WRITE (IPRINT,120)                                                RESU0029
      DO 20 I = 1, NPAR                                                 RESU0030
        IND = ISUB(I)                                                   RESU0031
        IF (ISTART(I)  LT  2) GO TO 10                                  RESU0032
        WRITE (IPRINT,130) I, IND, BETA(I)                              RESU0033
        GO TO 20                                                        RESU0034
   10   TSTAT = BETA(I) / COV(I,I)                                      RESU0035
        IF (IND  EQ  IDEP) WRITE (IPRINT,140) I, IND, BETA(I), COV(I,I),RESU0036
     1  TSTAT                                                           RESU0037
        IF (IND  NE  IDEP) WRITE (IPRINT,150) I, IND, BETA(I), COV(I,I),RESU0038
     1  TSTAT                                                           RESU0039
   20 CONTINUE                                                          RESU0040
C                                                                       RESU0041
C         ASYMPTOTIC QUANTITIES                                         RESU0042
      IF (MLE) WRITE (IPRINT,160) XLOGL                                 RESU0043
      IF (DET  LE  ZERO) WRITE (IPRINT,170)                             RESU0044
      IF (DET  GT  ZERO) WRITE (IPRINT,180) NACT, DET                   RESU0045
      WRITE (IPRINT,190)                                                RESU0046
      DO 30 I = 1, NPAR                                                 RESU0047
   30 WRITE (IPRINT,200) (COV(I,J),J=1,I)                               RESU0048
C                                                                       RESU0049
   40 FORMAT ('1MAXIMUM LIKELIHOOD BINARY ANALYSIS')                    RESU0050
   50 FORMAT ('1SELF-CRITICAL BINARY ANALYSIS, C =', D11 3)             RESU0051
   60 FORMAT ('OLOGISTIC TOLERANCE DISTRIBUTION')                       RESU0052
   70 FORMAT ('OGAUSSIAN TOLERANCE DISTRIBUTION')                       RESU0053
   80 FORMAT ('OEXTREME VALUE TOLERANCE DISTRIBUTION')                  RESU0054
   90 FORMAT ('OANALYSIS OF', I6, ' CASES AND', I4, ' VARIABLES'/'O',   RESU0055
     1        I4, ' PARAMETERS WERE ESTIMATED AND', I4,                 RESU0056
     2        ' WERE HELD CONSTANT', 2(/,'O'))                          RESU0057
  100 FORMAT ('OPARAMETERS ARE EXPRESSED IN REGRESSION FORM'/'O')       RESU0058
  110 FORMAT ('OPARAMETERS ARE EXPRESSED IN LOCATION-SCALE FORM'/'O')   RESU0059
  120 FORMAT ('     I  VAR', 8X, 'BETA(I)', 6X, 'STD  ERR ', 9X,        RESU0060
     1        'T STAT', 5X, 'STATUS')                                   RESU0061
  130 FORMAT (2I5, D15 7, 35X, 'FIXED LOCATION')                        RESU0062
  140 FORMAT (2I5, 3D15 7, 5X, 'ESTIMATED SCALE')                       RESU0063
  150 FORMAT (2I5, 3D15 7, 5X, 'ESTIMATED LOCATION')                    RESU0064
  160 FORMAT (2(/,'O'), /, ' THE LOG LIKELIHOOD IS', D16 8)             RESU0065
  170 FORMAT (2(/,'O'), /, ' THE ESTIMATED ASYMPTOTIC COVARIANCE',      RESU0066
     1        ' MATRIX IS NOT POSITIVE DEFINITE')                       RESU0067
  180 FORMAT (2(/,'O'), /, ' THE', I4, ' ROOT OF THE DETERMINANT ',     RESU0068
     1        'OF THE ESTIMATED'/' ASYMPTOTIC COVARIANCE MATRIX IS',    RESU0069
     2        D16 8)                                                    RESU0070
  190 FORMAT (/, 'OESTIMATED ASYMPTOTIC STD  ERRORS (ON DIAGONAL)',     RESU0071
     1        ', CORRELATIONS (BELOW DIAGONAL)'/' ')                    RESU0072
  200 FORMAT (10D12 4)                                                  RESU0073
      RETURN                                                            RESU0074
      END                                                               RESU0075
C***********************************************************************DLOG0001
C         FIRST AND SECOND PARTIAL DERIVATIVES FOR LOGISTIC BINARY,     DLOG0002
C         REGRESSION PARAMETERIZATION   PARTIALS ARE SCALED BY          DLOG0003
C         SAMPLE SIZE                                                   DLOG0004
```

```
          DDENS = XJI * EITHER                                          DLOG0065
          GRAD(J) = GRAD(J) + DDENS                                     DLOG0066
          DDENS = XJI * (S - F)                                         DLOG0067
          IF (J .EQ. 1) DDENS = DDENS + EXTRA                           DLOG0068
          DDENS = DDENS * FC - G * XJI                                  DLOG0069
          DO 70 K = J, NACT                                             DLOG0070
            IND = IACT(K)                                               DLOG0071
            HESS(K,J) = HESS(K,J) + X(IND,I) * DDENS                    DLOG0072
   70     CONTINUE                                                      DLOG0073
   80   CONTINUE                                                        DLOG0074
   90 CONTINUE                                                          DLOG0075
C                                                                       DLOG0076
C        SCALE GRADIENT AND HESSIAN BY SAMPLE SIZE                      DLOG0077
      EXTRA = DBLE(FLOAT(N))                                            DLOG0078
      DO 100 J = 1, NACT                                                DLOG0079
        GRAD(J) = GRAD(J) / EXTRA                                       DLOG0080
        DO 100 I = J, NACT                                              DLOG0081
          HESS(I,J) = HESS(I,J) / EXTRA                                 DLOG0082
          HESS(J,I) = HESS(I,J)                                         DLOG0083
  100 CONTINUE                                                          DLOG0084
      IFAULT = 0                                                        DLOG0085
      RETURN                                                            DLOG0086
C                                                                       DLOG0087
C        ERROR EXIT                                                     DLOG0088
  110 IFAULT = -1                                                       DLOG0089
C                                                                       DLOG0090
      RETURN                                                            DLOG0091
      END                                                               DLOG0092
C***********************************************************************VLOG0001
C        COMPUTE V(*,*) FACTOR FOR ASYMPTOTIC COVARIANCE MATRIX,        VLOG0002
C        LOGISTIC BINARY.  CALLED ONLY WHEN C .NE. 0                    VLOG0003
C***********************************************************************VLOG0004
      SUBROUTINE VLOGBN(C, NACT, IACT, PAR, V, N, IX, X, IA)            VLOG0005
C        ARGUMENTS                                                      VLOG0006
      INTEGER NACT, IACT(NACT), N, IX, IA(N)                            VLOG0007
      REAL X(IX,N)                                                      VLOG0008
      DOUBLE PRECISION C, PAR(NACT), V(NACT,NACT)                       VLOG0009
C        LOCAL SCALARS                                                  VLOG0010
      INTEGER IND                                                       VLOG0011
      DOUBLE PRECISION FC, S, F, EITHER, TEMP, ZERO, ONE, BIG           VLOG0012
C        MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT        VLOG0013
C        DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG                   VLOG0014
      DATA ZERO, ONE, BIG /0.0D0, 1.0D0, 174.0D0/                       VLOG0015
C                                                                       VLOG0016
C        SET UPPER TRIANGLE OF V(*,*) TO 0                              VLOG0017
      DO 10 J = 1, NACT                                                 VLOG0018
        DO 10 I = J, NACT                                               VLOG0019
   10 V(I,J) = ZERO                                                     VLOG0020
C                                                                       VLOG0021
C        MAIN LOOP OVER SAMPLE                                          VLOG0022
      DO 50 I = 1, N                                                    VLOG0023
        FC = ZERO                                                       VLOG0024
        DO 20 J = 1, NACT                                               VLOG0025
          IND = IACT(J)                                                 VLOG0026
          FC = FC + PAR(J) * X(IND,I)                                   VLOG0027
   20   CONTINUE                                                        VLOG0028
        IF (DABS(FC) .GE. BIG) GO TO 50                                 VLOG0029
        FC = DEXP(FC)                                                   VLOG0030
        S = ONE + FC                                                    VLOG0031
        F = FC / S                                                      VLOG0032
```

```
      S = ONE / S                                                       VLOG0033
      FC = (F*S) ** C                                                   VLOG0034
      EITHER = S                                                        VLOG0035
      IF (IA(I) .EQ. 0) EITHER = F                                      VLOG0036
      EITHER = (EITHER*FC) ** 2                                         VLOG0037
C                                                                       VLOG0038
C        INCREMENT TERM OF V(*,*)                                       VLOG0039
      DO 40 J = 1, NACT                                                 VLOG0040
        IND = IACT(J)                                                   VLOG0041
        TEMP = EITHER * X(IND,I)                                        VLOG0042
        DO 30 K = J, NACT                                               VLOG0043
          IND = IACT(K)                                                 VLOG0044
          V(K,J) = V(K,J) + TEMP * X(IND,I)                             VLOG0045
   30   CONTINUE                                                        VLOG0046
   40 CONTINUE                                                          VLOG0047
   50 CONTINUE                                                          VLOG0048
C                                                                       VLOG0049
C        DIVIDE V(*,*) BY SAMPLE SIZE, FILL OUT                         VLOG0050
      TEMP = DBLE(FLOAT(N))                                             VLOG0051
      DO 60 J = 1, NACT                                                 VLOG0052
        DO 60 I = J, NACT                                               VLOG0053
          V(I,J) = V(I,J) / TEMP                                        VLOG0054
          V(J,I) = V(I,J)                                               VLOG0055
   60 CONTINUE                                                          VLOG0056
C                                                                       VLOG0057
      RETURN                                                            VLOG0058
      END                                                               VLOG0059
C***********************************************************************DGAU0001
C        FIRST AND SECOND PARTIAL DERIVATIVES FOR BINARY GAUSSIAN,      DGAU0002
C        REGRESSION PARAMETERIZATION.  PARTIALS ARE SCALED BY           DGAU0003
C        SAMPLE SIZE                                                    DGAU0004
C        FAILURE CODE - IFAULT = -I - AN EXCEPTION WAS ABOUT TO         DGAU0005
C                OCCUR WHILE PROCESSING THE I TH OBSERVATION            DGAU0006
C***********************************************************************DGAU0007
      SUBROUTINE DGAUBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX,  DGAU0008
     1           X, IA, IFAULT)                                         DGAU0009
C        ARGUMENTS                                                      DGAU0010
      INTEGER NACT, IACT(NACT), N, IX, IA(N), IFAULT                    DGAU0011
      REAL X(IX,N)                                                      DGAU0012
      DOUBLE PRECISION C, OBJECT, PAR(NACT), GRAD(NACT), HESS(NACT,NACT)DGAU0013
C        LOCAL SCALARS                                                  DGAU0014
      LOGICAL MLE                                                       DGAU0015
      INTEGER IND                                                       DGAU0016
      DOUBLE PRECISION EXTRA, DOT, DOTMIN, FC, CBY2, RATIO, EITHER,     DGAU0017
     1    HTERM, XJI, DDENS, ZERO, ONE, TWO, BIG                        DGAU0018
C        FUNCTIONS CALLED                                               DGAU0019
      DOUBLE PRECISION ALNORM, RMILLS                                   DGAU0020
C        MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT        DGAU0021
C        DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG                   DGAU0022
      DATA ZERO, ONE, TWO, BIG /0.0D0, 1.0D0, 2.0D0, 174.0D0/           DGAU0023
C                                                                       DGAU0024
      IFAULT = 0                                                        DGAU0025
      MLE = ONE + C .EQ. ONE                                            DGAU0026
      EXTRA = ONE / PAR(1)                                              DGAU0027
      CBY2 = C / TWO                                                    DGAU0028
      OBJECT = ZERO                                                     DGAU0029
      DO 10 J = 1, NACT                                                 DGAU0030
        GRAD(J) = ZERO                                                  DGAU0031
        DO 10 I = J, NACT                                               DGAU0032
          HESS(I,J) = ZERO                                              DGAU0033
```

```
   10 CONTINUE
C
C        MAIN LOOP OVER SAMPLE
      DO 90 I = 1, N
        DOT = ZERO
        DO 20 J = 1, NACT
          IND = IACT(J)
          DOT = DOT + PAR(J) * X(IND,I)
   20   CONTINUE
        DOTMIN = -DOT
        IF (IA(I) .EQ. 0) GO TO 30
        RATIO = RMILLS(DOTMIN)
        EITHER = RATIO
        GO TO 40
   30   RATIO = RMILLS(DOT)
        DOT = DOTMIN
        EITHER = -RATIO
   40   IF (MLE) GO TO 50
        XJI = CBY2 * DOT * DOT
        IF (DABS(XJI) .GT. BIG) GO TO 110
        FC = DEXP(-XJI)
        GO TO 60
   50   FC = ONE
        XJI = ALNORM(DOT, .FALSE. )
        IF (XJI .EQ. ZERO) GO TO 110
        OBJECT = OBJECT + DLOG(XJI)
C
C        INCREMENT GRADIENT AND HESSIAN
   60   EITHER = FC * EITHER
        HTERM = -FC * RATIO * (DOT + RATIO)
        FC = C * EITHER
        DO 80 J = 1, NACT
          IND = IACT(J)
          XJI = DBLE(X(IND,I))
          GRAD(J) = GRAD(J) + XJI * EITHER
          DDENS = XJI * DOTMIN
          IF (J .EQ. 1) DDENS = DDENS + EXTRA
          DDENS = DDENS * FC + XJI * HTERM
          DO 70 K = J, NACT
            IND = IACT(K)
            HESS(K,J) = HESS(K,J) + X(IND,I) * DDENS
   70     CONTINUE
   80   CONTINUE
   90 CONTINUE
C
C        SCALE GRADIENT AND HESSIAN BY SAMPLE SIZE
      EXTRA = DBLE(FLOAT(N))
      DO 100 J = 1, NACT
        GRAD(J) = GRAD(J) / EXTRA
        DO 100 I = J, NACT
          HESS(I,J) = HESS(I,J) / EXTRA
          HESS(J,I) = HESS(I,J)
  100 CONTINUE
      IFAULT = 0
      RETURN
C
C        ERROR EXIT
  110 IFAULT = -I
C
      RETURN
```

```
      END                                                               DGAU0094
C***********************************************************************VGAU0001
C         COMPUTE V(*,*) FACTOR FOR ASYMPTOTIC COVARIANCE MATRIX,        VGAU0002
C         GAUSSIAN BINARY.  CALLED ONLY WHEN C .NE. O                    VGAU0003
C***********************************************************************VGAU0004
      SUBROUTINE VGAUBN(C, NACT, IACT, PAR, V, N, IX, X, IA)             VGAU0005
C         ARGUMENTS                                                      VGAU0006
      INTEGER NACT, IACT(NACT), N, IX, IA(N)   .                         VGAU0007
      REAL X(IX,N)                                                       VGAU0008
      DOUBLE PRECISION C, PAR(NACT), V(NACT,NACT)                        VGAU0009
C         LOCAL SCALARS                                                  VGAU0010
      INTEGER IND                                                        VGAU0011
      DOUBLE PRECISION F2C, TEMP, EITHER, ZERO, BIG                      VGAU0012
C         FUNCTION CALLED                                                VGAU0013
      DOUBLE PRECISION RMILLS                                            VGAU0014
C         MACHINE-DEPENDENT CONSTANT - BIG ROUGHLY CHOSEN SO THAT        VGAU0015
C         DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG                   VGAU0016
      DATA ZERO, BIG /0.0D0, 174.0D0/                                    VGAU0017
C                                                                        VGAU0018
C         SET UPPER TRIANGLE OF V(*,*) TO ZERO                           VGAU0019
      DO 10 J = 1, NACT                                                  VGAU0020
        DO 10 I = J, NACT                                                VGAU0021
   10 V(I,J) = ZERO                                                      VGAU0022
C                                                                        VGAU0023
C         MAIN LOOP OVER SAMPLE                                          VGAU0024
      DO 70 I = 1, N                                                     VGAU0025
        F2C = ZERO                                                       VGAU0026
        DO 20 J = 1, NACT                                                VGAU0027
          IND = IACT(J)                                                  VGAU0028
          F2C = F2C + PAR(J) * X(IND,I)                                  VGAU0029
   20   CONTINUE                                                         VGAU0030
        TEMP = C * F2C * F2C                                             VGAU0031
        IF (DABS(TEMP) .GE. BIG) GO TO 70                                VGAU0032
        IF (IA(I) .EQ. O) GO TO 30                                       VGAU0033
        EITHER = RMILLS(-F2C)                                            VGAU0034
        GO TO 40                                                         VGAU0035
   30   EITHER = RMILLS(F2C)                                             VGAU0036
   40   F2C = DEXP(-TEMP) * EITHER * EITHER                              VGAU0037
        DO 60 J = 1, NACT                                                VGAU0038
          IND = IACT(J)                                                  VGAU0039
          TEMP = F2C * X(IND,I)                                          VGAU0040
          DO 50 K = J, NACT                                              VGAU0041
            IND = IACT(K)                                                VGAU0042
            V(K,J) = V(K,J) + TEMP * X(IND,I)                            VGAU0043
   50     CONTINUE                                                       VGAU0044
   60   CONTINUE                                                         VGAU0045
   70 CONTINUE                                                           VGAU0046
C                                                                        VGAU0047
C         DIVIDE V(*,*) BY SAMPLE SIZE, FILL OUT                         VGAU0048
      TEMP = DBLE(FLOAT(N))                                              VGAU0049
      DO 80 J = 1, NACT                                                  VGAU0050
        DO 80 I = J, NACT                                                VGAU0051
          V(I,J) = V(I,J) / TEMP                                         VGAU0052
          V(J,I) = V(I,J)                                                VGAU0053
   80 CONTINUE                                                           VGAU0054
C                                                                        VGAU0055
      RETURN                                                             VGAU0056
      END                                                                VGAU0057
C***********************************************************************DEXV0001
C         FIRST AND SECOND PARTIAL DERIVATIVES FOR BINARY EXTREME        DEXV0002
```

```
C        VALUE, REGRESSION PARAMETERIZATION.  PARTIALS ARE SCALED       DEXV0003
C        BY SAMPLE SIZE.                                                DEXV0004
C        FAILURE CODE - IFAULT = -I - AN EXCEPTION WOULD HAVE           DEXV0005
C               OCCURRED WHEN PROCESSING THE I TH OBSERVATION           DEXV0006
C***********************************************************************DEXV0007
      SUBROUTINE DEXVBN(C, OBJECT, NACT, IACT, PAR, GRAD, HESS, N, IX,  DEXV0008
     1          X, IA, IFAULT)                                          DEXV0009
C        ARGUMENTS                                                      DEXV0010
      INTEGER NACT, IACT(NACT), N, IX, IA(N), IFAULT                    DEXV0011
      REAL X(IX,N)                                                      DEXV0012
      DOUBLE PRECISION C, OBJECT, PAR(NACT), GRAD(NACT), HESS(NACT,NACT)DEXV0013
C        LOCAL SCALARS                                                  DEXV0014
      LOGICAL MLE                                                       DEXV0015
      INTEGER IND                                                       DEXV0016
      DOUBLE PRECISION EXTRA, DOT, EXPON, DENFAC, EXPEXP, FC, EITHER,   DEXV0017
     1       HTERM, ZERO, ONE, BIG, BIG1                                DEXV0018
C        MACHINE-DEPENDENT CONSTANTS - BIG ROUGHLY CHOSEN SO THAT       DEXV0019
C        DEXP(X) WILL CAUSE EXCEPTION IF /X/ .GT. BIG, BIG1 SUCH        DEXV0020
C        THAT 1.0D0 / DEXP(DEXP(X)) .NE. 1.0D0 IF X .GT. BIG1           DEXV0021
      DATA ZERO, ONE, BIG, BIG1 /0.0D0, 1.0D0, 174.0D0, -36.2D0/        DEXV0022
C                                                                       DEXV0023
      IFAULT = 0                                                        DEXV0024
      MLE = ONE + C .EQ. ONE                                            DEXV0025
      EXTRA = ONE / PAR(1)                                              DEXV0026
      OBJECT = ZERO                                                     DEXV0027
      DO 10 J = 1, NACT                                                 DEXV0028
        GRAD(J) = ZERO                                                  DEXV0029
        DO 10 I = J, NACT                                               DEXV0030
          HESS(I,J) = ZERO                                              DEXV0031
   10 CONTINUE                                                          DEXV0032
C                                                                       DEXV0033
C        MAIN LOOP OVER SAMPLE                                          DEXV0034
      DO 90 I = 1, N                                                    DEXV0035
        DOT = ZERO                                                      DEXV0036
        DO 20 J = 1, NACT                                               DEXV0037
          IND = IACT(J)                                                 DEXV0038
          DOT = DOT + PAR(J) * X(IND,I)                                 DEXV0039
   20   CONTINUE                                                        DEXV0040
        IF (DABS(DOT) .GE. BIG) GO TO 110                               DEXV0041
        EXPON = DEXP(DOT)                                               DEXV0042
        DENFAC = ONE - EXPON                                            DEXV0043
        IF (IA(I) .EQ. 0) GO TO 40                                      DEXV0044
C                                                                       DEXV0045
C        FAILURE TERM, IA(I) = 1                                        DEXV0046
        IF (EXPON .GE. BIG .OR. DOT .LE. BIG1) GO TO 110                DEXV0047
        EXPEXP = DEXP(EXPON)                                            DEXV0048
        DOT = ONE - ONE / EXPEXP                                        DEXV0049
        FC = EXPON / EXPEXP                                             DEXV0050
        EITHER = FC / DOT                                               DEXV0051
        HTERM = EITHER * (DENFAC - EITHER)                              DEXV0052
        IF (MLE) GO TO 30                                               DEXV0053
        FC = FC ** C                                                    DEXV0054
        GO TO 60                                                        DEXV0055
   30   FC = ONE                                                        DEXV0056
        OBJECT = OBJECT + DLOG(DOT)                                     DEXV0057
        GO TO 60                                                        DEXV0058
C                                                                       DEXV0059
C        NON FAILURE TERM, IA(I) = 0                                    DEXV0060
   40   EITHER = -EXPON                                                 DEXV0061
        HTERM = EITHER                                                  DEXV0062
```

```
      IF (MLE) GO TO 50                                                 DEXV0063
      FC = C * (DOT - EXPON)                                            DEXV0064
      IF (DABS(FC)  GE  BIG) GO TO 110                                  DEXV0065
      FC = DEXP(FC)                                                     DEXV0066
      GO TO 60                                                          DEXV0067
   50 FC = ONE                                                          DEXV0068
      OBJECT = OBJECT + EITHER                                          DEXV0069
C                                                                       DEXV0070
C        LOOP TO INCREMENT THE DERIVATIVES                              DEXV0071
   60 EITHER = FC * EITHER                                              DEXV0072
      HTERM = FC * HTERM                                                DEXV0073
      FC = C * EITHER                                                   DEXV0074
      DO 80 J = 1, NACT                                                 DEXV0075
        IND = IACT(J)                                                   DEXV0076
        DOT = DBLE(X(IND,I))                                            DEXV0077
        GRAD(J) = GRAD(J) + DOT * EITHER                                DEXV0078
        EXPON = DOT * DENFAC                                            DEXV0079
        IF (J  EQ  1) EXPON = EXPON + EXTRA                             DEXV0080
        EXPON = FC * EXPON + DOT * HTERM                                DEXV0081
        DO 70 K = J, NACT                                               DEXV0082
          IND = IACT(K)                                                 DEXV0083
          HESS(K,J) = HESS(K,J) + X(IND,I) * EXPON                      DEXV0084
   70   CONTINUE                                                        DEXV0085
   80 CONTINUE                                                          DEXV0086
   90 CONTINUE                                                          DEXV0087
C                                                                       DEXV0088
C        SCALE PARTIALS BY SAMPLE SIZE, NORMAL EXIT                     DEXV0089
      EXTRA = DBLE(FLOAT(N))                                            DEXV0090
      DO 100 J = 1, NACT                                                DEXV0091
        GRAD(J) = GRAD(J) / EXTRA                                       DEXV0092
        DO 100 I = J, NACT                                              DEXV0093
          HESS(I,J) = HESS(I,J) / EXTRA                                 DEXV0094
          HESS(J,I) = HESS(I,J)                                         DEXV0095
  100 CONTINUE                                                          DEXV0096
      RETURN                                                            DEXV0097
C                                                                       DEXV0098
C        ERROR EXIT - EXCEPTIONS                                        DEXV0099
  110 IFAULT = -I                                                       DEXV0100
C                                                                       DEXV0101
      RETURN                                                            DEXV0102
      END                                                               DEXV0103
C***********************************************************************VEXV0001
C        COMPUTE V(*,*) FACTOR FOR ASYMPTOTIC COVARIANCE MATRIX,        VEXV0002
C        BINARY EXTREME VALUE   CALLED ONLY WHEN C  NE  O               VEXV0003
C***********************************************************************VEXV0004
      SUBROUTINE VEXVBN(C, NACT, IACT, PAR, V, N, IX, X, IA)            VEXV0005
C        ARGUMENTS                                                      VEXV0006
      INTEGER NACT, IACT(NACT), N, IX, IA(N)                            VEXV0007
      REAL X(IX,N)                                                      VEXV0008
      DOUBLE PRECISION C, PAR(NACT), V(NACT,NACT)                       VEXV0009
C        LOCAL SCALARS                                                  VEXV0010
      INTEGER IND                                                       VEXV0011
      DOUBLE PRECISION DOT, EXPON, FC, ZERO, ONE, BIG, BIG1             VEXV0012
C        MACHINE-DEPENDENT CONSTANTS   BIG ROUGHLY CHOSEN SO THAT       VEXV0013
C        DEXP(X) WILL CAUSE EXCEPTION IF /X/  GT  BIG, BIG1 SUCH        VEXV0014
C        THAT 1 0D0 / DEXP(DEXP(X))  NE  1 0D0 IF X  GT  BIG1           VEXV0015
      DATA ZERO, ONE, BIG, BIG1 /0 0D0, 1 0D0, 174 0D0, -36 2D0/        VEXV0016
C                                                                       VEXV0017
C        SET UPPER TRIANGLE OF V(*,*) TO ZERO                           VEXV0018
      DO 10 J = 1, NACT                                                 VEXV0019
```

```
      DO 10 I = J, NACT
   10 V(I,J) = ZERO
C
C        MAIN LOOP OVER SAMPLE
      DO 70 I = 1, N
        DOT = ZERO
        DO 20 J = 1, NACT
          IND = IACT(J)
          DOT = DOT + PAR(J) * X(IND,I)
   20   CONTINUE
        IF (DABS(DOT) .GE. BIG) GO TO 70
        EXPON = DEXP(DOT)
        IF (IA(I) .EQ. O) GO TO 30
        IF (DOT .LT. BIG1) GO TO 70
        FC = DEXP(EXPON)
        DOT = ONE - ONE / FC
        FC = EXPON / FC
        EXPON = FC / DOT
        FC = EXPON * FC ** C
        GO TO 40
   30   FC = C * (DOT - EXPON)
        IF (DABS(FC) .GT. BIG) GO TO 70
        FC = EXPON * DEXP(FC)
C
C        SUMMANDS FOR V(*,*)
   40   FC = FC * FC
        DO 60 J = 1, NACT
          IND = IACT(J)
          DOT = FC * X(IND,I)
          DO 50 K = J, NACT
            IND = IACT(K)
            V(K,J) = V(K,J) + DOT * X(IND,I)
   50     CONTINUE
   60   CONTINUE
   70 CONTINUE
C
C        DIVIDE V(*,*) BY SAMPLE SIZE, FILL OUT
      DOT = DBLE(FLOAT(N))
      DO 80 J = 1, NACT
        DO 80 I = J, NACT
          V(I,J) = V(I,J) / DOT
          V(J,I) = V(I,J)
   80 CONTINUE
C
      RETURN
      END
C*********************************************************************
C        DOUBLE PRECISION MATRIX MULTIPLICATION
C        X(N1 BY N3) = Y(N1 BY N2) * Z(N2 BY N3)
C        THREE OPTIONS -
C              IFLAG = O - X, Y, AND Z ARE DISTINCT
C              IFLAG .LT. O - X, Y OVERLAP, CORNER OF Y OVERWRITTEN
C              IFLAG .GT. O - X, Z OVERLAP, CORNER OF Z OVERWRITTEN
C*********************************************************************
      SUBROUTINE DMXMLT(X, IX, N1, Y, IY, N2, Z, IZ, N3, WORK, LWORK,
     1           IFLAG, IFAULT)
C        ARGUMENTS
      INTEGER IX, N1, IY, N2, IZ, N3, LWORK, IFLAG, IFAULT
      DOUBLE PRECISION X(IX,N3), Y(IY,N2), Z(IZ,N3), WORK(LWORK)
C        LOCAL SCALARS
```

```
      DOUBLE PRECISION TEMP, ZERO
      DATA ZERO /0 0D0/
C
C        ERROR EXITS
      IFAULT = 1
      IF (MINO(N1,N2,N3) .LT  1  OR  MINO(IX,IY)  LT. N1 .OR. IZ .LT.
     1      N2) RETURN
      IFAULT = 2
      IF (IFLAG  LT  0  AND  (N3  GT  N2 .OR. LWORK .LT. N3))
     1      RETURN
      IFAULT = 3
      IF (IFLAG  GT. 0  AND  (N1  GT. IZ .OR. LWORK .LT. N1))
     1      RETURN
      IFAULT = 0
      IF (IFLAG  NE  0) GO TO 30
C
C        STRAIGHTFORWARD, NO OVERWRITING
      DO 20 I = 1, N1
        DO 20 J = 1, N3
          TEMP = ZERO
          DO 10 K = 1, N2
   10     TEMP = TEMP + Y(I,K) * Z(K,J)
          X(I,J) = TEMP
   20 CONTINUE
      RETURN
C
C        CORNER OF MATRIX Z IS OVERWRITTEN
   30 IF (IFLAG  LT  0) GO TO 80
      DO 70 J = 1, N3
        DO 50 I = 1, N1
          TEMP = ZERO
          DO 40 K = 1, N2
   40     TEMP = TEMP + Y(I,K) * Z(K,J)
          WORK(I) = TEMP
   50   CONTINUE
        DO 60 I = 1, N1
   60   X(I,J) = WORK(I)
   70 CONTINUE
      RETURN
C
C        CORNER OF MATRIX Y IS OVERWRITTEN
   80 DO 120 I = 1, N1
        DO 100 J = 1, N3
          TEMP = ZERO
          DO 90 K = 1, N2
   90     TEMP = TEMP + Y(I,K) * Z(K,J)
          WORK(J) = TEMP
  100   CONTINUE
        DO 110 J = 1, N3
  110   X(I,J) = WORK(J)
  120 CONTINUE
C
      RETURN
      END
C*****************************************************************
C        DOUBLE PRECISION VERSION OF ALGORITHM AS 66, APPLIED
C        STATISTICS (1973), VOL  22, NO  3
C        EVALUATES THE TAIL AREA OF THE STANDARD NORMAL CURVE
C        FROM X TO INFINITY IF UPPER IS  TRUE  OR FROM
C        MINUS INFINITY TO X IF UPPER IS  FALSE
```

```
C***********************************************************************
      DOUBLE PRECISION FUNCTION ALNORM(X,UPPER)
C        ARGUMENTS
      LOGICAL UPPER
      DOUBLE PRECISION X
C        LOCAL SCALARS
      LOGICAL UP
      DOUBLE PRECISION LTONE, UTZERO, ZERO, HALF, ONE, CON, Z, Y
C        MACHINE-DEPENDENT CONSTANTS - LTONE = (N + 9) / 3, WHERE
C        N IS NO. OF DECIMAL DIGITS IN DOUBLE PRECISION NUMBER,
C        UTZERO SUCH THAT DEXP(-X*X/2.0D0) WILL CAUSE AN EXCEPTION
C        IF X .GT. UTZERO.
C        CONSTANTS IN EXPRESSIONS ARE AS IN AS 66.
      DATA LTONE, UTZERO, ZERO, HALF, ONE, CON /8.0D0, 18.65D0, 0.0D0,
     1     0.5D0, 1.0D0, 1.28D0/
C
      UP = UPPER
      Z = X
      IF (Z .GE. ZERO) GO TO 10
      UP = .NOT. UP
      Z = -Z
   10 IF (Z .LE. LTONE .OR. UP .AND. Z .LE. UTZERO) GO TO 20
      ALNORM = ZERO
      GO TO 40
   20 Y = HALF * Z * Z
      IF (Z .GT. CON) GO TO 30
C
      ALNORM = HALF - Z * (0.398942280444D0-0.399903438504D0*Y/(Y + 5.
     1758854804580D0-29.821355780800/(Y + 2.62433121679D0+48.
     26959930692D0/(Y + 5.92885724438D0))))
      GO TO 40
C
   30 ALNORM = 0.398942280385D0 * DEXP(-Y) / (Z - 3.8052D-8 + 1.
     100000615302D0/(Z + 3.98064794D-4 + 1.98615381364D0/(Z - 0.
     215167911663500+5.29330324926D0/(Z + 4.83859128080D0-15.
     3150897245100/(Z + 0.742380924027D0+30.789933034D0/(Z + 3.
     4990194170110D0)))))))
C
   40 IF ( .NOT. UP) ALNORM = ONE - ALNORM
C
      RETURN
      END
C***********************************************************************
C        RECIPROCAL OF MILLS RATIO, Z(X) / Q(X), X A STANDARD NORMAL
C        VARIATE.  BASED ON FUNCTION ALNORM(), ALGORITHM AS 66.
C***********************************************************************
      DOUBLE PRECISION FUNCTION RMILLS(X)
C        ARGUMENTS
      DOUBLE PRECISION X
C        LOCAL SCALARS
      LOGICAL UP
      DOUBLE PRECISION Z, Y, LTONE, UTZERO, ZERO, HALF, ONE, CON, FPI1,
     1     FPI2
C        MACHINE-DEPENDENT CONSTANTS - LTONE = (N + 9) / 3, WHERE
C        N IS NO. OF DECIMAL DIGITS IN DOUBLE PRECISION NUMBER,
C        UTZERO SUCH THAT DEXP(-X*X/2.0D0) WILL CAUSE AN EXCEPTION
C        IF X .LT. UTZERO.
C        FPI1 = SQRT(2/PI), FPI2 = 1/SQRT(2*PI)
C        CONSTANTS IN EXPRESSIONS ARE AS IN AS 66.
      DATA LTONE, UTZERO, ZERO, HALF, ONE, CON, FPI1, FPI2 /8.0D0,
```

```
     1     -18.65D0, 0.0D0, 0.5D0, 1.0D0, 1.28D0, 7.97884560802865360D-1,RMIL0019
     2      3.98942280401432680D-1/                                     RMIL0020
C                                                                       RMIL0021
C        TRIVIAL CASES                                                  RMIL0022
      IF (X .NE. ZERO) GO TO 10                                         RMIL0023
      RMILLS = FPI1                                                     RMIL0024
      RETURN                                                            RMIL0025
   10 IF (X .GT. UTZERO) GO TO 20                                       RMIL0026
      RMILLS = ZERO                                                     RMIL0027
      RETURN                                                            RMIL0028
C                                                                       RMIL0029
C        USUAL SITUATION                                                RMIL0030
   20 UP = .TRUE.                                                       RMIL0031
      Z = X                                                             RMIL0032
      IF (Z .GE. ZERO) GO TO 30                                         RMIL0033
      UP = .FALSE.                                                      RMIL0034
      Z = -Z                                                            RMIL0035
   30 Y = HALF * Z * Z                                                  RMIL0036
      IF (Z .GT. CON) GO TO 50                                          RMIL0037
C                                                                       RMIL0038
C        CENTRAL PORTION - 0 .LT. ABS(X) .LT. 1.28                      RMIL0039
      RMILLS = Z * (0.398942280444D0-0.399903438504D0*Y/(Y + 5.         RMIL0040
     1758854804580D0-29.8213557808D0/(Y + 2.62433121679D0+48.           RMIL0041
     26959930692D0/(Y + 5.92885724438D0))))                             RMIL0042
      Y = FPI2 * DEXP(-Y)                                               RMIL0043
      IF (UP) GO TO 40                                                  RMIL0044
      RMILLS = Y / (HALF + RMILLS)                                      RMIL0045
      RETURN                                                            RMIL0046
   40 RMILLS = Y / (HALF - RMILLS)                                      RMIL0047
      RETURN                                                            RMIL0048
C                                                                       RMIL0049
C        OUTER PORTION - ABS(X) .GE. 1.28                               RMIL0050
   50 IF (UP .OR. Z .LE. LTONE) GO TO 60                                RMIL0051
C                                                                       RMIL0052
C        SPECIAL CASE - LOWER TAIL AND Q ESSENTIALLY 1                  RMIL0053
      RMILLS = FPI2 * DEXP(-Y)                                          RMIL0054
      RETURN                                                            RMIL0055
C                                                                       RMIL0056
C        USUAL SITUATION                                                RMIL0057
   60 RMILLS = (Z - 3.8052D-8 + 1.00000615302D0/(Z + 3.98064794D-4 + 1. RMIL0058
     198615381364D0/(Z - 0.151679116635D0+5.29330324926D0/(Z + 4.       RMIL0059
     28385912808D0-15.150897245 1D0/(Z + 0.742380924027D0+30.789933034D0/RMIL0060
     3(Z + 3.99019417011D0))))))                                        RMIL0061
C                                                                       RMIL0062
      IF (UP) RETURN                                                    RMIL0063
      Y = FPI2 * DEXP(-Y)                                               RMIL0064
      RMILLS = Y / (ONE - Y/RMILLS)                                     RMIL0065
C                                                                       RMIL0066
      RETURN                                                            RMIL0067
      END                                                               RMIL0068
```

```
C*********************************************************************
C        SAMPLE MAIN PROGRAM AND INPUT SUBROUTINE FOR SELF-CRITICAL
C        BINARY ESTIMATION.  IT IS RECOMMENDED THAT BINARY() AND
C        ITS AUXILIARY PROCEDURES BE COMPILED AND PLACED IN AN
C        OBJECT CODE LIBRARY, SO THE USER CAN CALL BINARY() FROM
C        ARBITRARY PROGRAMS.
C*********************************************************************
      INTEGER N, IX, IA(750), NPAR, ISUB(10), ISTART(10), IDEP, IDIST,
     1        MAXIT, IPRINT, IFLAG, ICOV, LMEM, MEMORY(490), IFAULT, NC,
     2        NMAX, NCMAX, IREAD
      REAL X(10,750)
      DOUBLE PRECISION C(5), RELTOL, ABSTOL, BETA(10), XLOGL, COV(10,10)
C
C        DIMENSION SPECIFICATIONS FOR ARRAYS.
C        THIS MAIN PROGRAM CAN HANDLE UP TO 750 OBSERVATIONS,
C        WITH UP TO 10 PARAMETERS AND 5 DIFFERENT VALUES OF
C        C FOR ESTIMATION.
      DATA IX, ICOV, LMEM, NMAX, NCMAX /10, 10, 490, 750, 5/
C
C        BASIC INFORMATION TO PASS TO BINARY() - IPRINT STANDS
C        FOR LOGICAL OUTPUT UNIT 6
      DATA MAXIT, RELTOL, ABSTOL, IPRINT, IFLAG /15, 1.0D-7, 1.0D-7, 6,
     1     0/
C
C        IREAD IS LOGICAL INPUT UNIT 5
      DATA IREAD /5/
C
C        SPECIFY EVERYTHING BY HAND IN SUBROUTINE INPUT - ARRAY
C        MEMORY(*) PASSED AS WORKSPACE
      CALL INPUT(IX, N, NMAX, X, IA, NPAR, BETA, ISUB, ISTART, NC,
     1     NCMAX, C, IDEP, IDIST, MEMORY, IREAD, IPRINT, IFAULT)
      IF (IFAULT .EQ. 0) GO TO 10
      IF (IFAULT .EQ. 1) WRITE (IPRINT,50) N, NMAX
      IF (IFAULT .EQ. 2) WRITE (IPRINT,60) NPAR, IX
      IF (IFAULT .EQ. 3) WRITE (IPRINT,70) NC, NCMAX
      WRITE (IPRINT,80)
      STOP
C
C        LOOP OVER THE VALUES OF C REQUESTED FOR ESTIMATION
   10 DO 40 I = 1, NC
        CALL BINARY(N, IX, X, IA, NPAR, ISUB, ISTART, IDEP, IDIST, C(I),
     1       RELTOL, ABSTOL, MAXIT, IPRINT, IFLAG, BETA, XLOGL, ICOV,
     2       COV, LMEM, MEMORY, IFAULT)
        IF (IFAULT .EQ. 0) GO TO 20
        WRITE (IPRINT,90) IFAULT
        STOP
   20   IF (I .GT. 1) GO TO 40
C
C        AFTER THE FIRST ESTIMATION, SET ISTART(*) TO 1, SO
C        LATEST ESTIMATES CAN BE USED AS STARTING VALUES
        DO 30 J = 1, NPAR
   30   ISTART(J) = 1
C
   40 CONTINUE
C
   50 FORMAT ('OERROR - SAMPLE SIZE OF', I10, ' EXCEEDS DIMENSION OF',
     1        I10)
   60 FORMAT ('OERROR - ', I3, ' PARAMETERS EXCEEDS DIMENSION OF', I3)
   70 FORMAT ('OERROR - ', I3, ' C VALUES EXCEEDS DIMENSION OF', I3)
   80 FORMAT ('ORECOMPILE MAIN PROGRAM WITH NEW DIMENSIONS')
```

```
   90 FORMAT ('OESTIMATION PROCEDURE HAS FAILED - IFAULT =', I6)       MAIN0061
      STOP                                                              MAIN0062
      END                                                               MAIN0063
C*********************************************************************INPU0001
C          CRUDE INPUT ROUTINE   SETS ESTIMATION CONTROL PARAMETERS     INPU0002
C          EXPLICITLY, INSTEAD OF READING THEM IN                       INPU0003
C*********************************************************************INPU0004
      SUBROUTINE INPUT(IX, N, NMAX, X, IA, NPAR, BETA, ISUB, ISTART, NC,INPU0005
     1               NCMAX, C, IDEP, IDIST, WORK, IREAD, IPRINT, IFAULT)INPU0006
C          ARGUMENTS                                                    INPU0007
      INTEGER IX, N, NMAX, IA(1), NPAR, ISUB(1), ISTART(1), NC, NCMAX,  INPU0008
     1        IDEP, IDIST, IPRINT, IFAULT                               INPU0009
      REAL X(IX,1), WORK(1)                                             INPU0010
      DOUBLE PRECISION BETA(1), C(1)                                    INPU0011
C          LOCAL TYPE DECLARATION                                       INPU0012
      LOGICAL FLAG                                                      INPU0013
C                                                                       INPU0014
C          SET DETAILS OF PROBLEM SIZE                                  INPU0015
      N = 680                                                           INPU0016
      NPAR = 5                                                          INPU0017
      NC = 4                                                            INPU0018
C                                                                       INPU0019
C          CHECK FOR SIZE ERRORS                                        INPU0020
      IFAULT = 1                                                        INPU0021
      IF (N  GT  NMAX) RETURN                                           INPU0022
      IFAULT = 2                                                        INPU0023
      IF (NPAR  GT  IX) RETURN                                          INPU0024
      IFAULT = 3                                                        INPU0025
      IF (NC  GT  NCMAX) RETURN                                         INPU0026
      IFAULT = 0                                                        INPU0027
C                                                                       INPU0028
C          MODELING DETAILS                                             INPU0029
C          DEPENDENT VARIABLE IN FIRST ROW                              INPU0030
      IDEP = 1                                                          INPU0031
C          CONSTANT IN SECOND ROW                                       INPU0032
      ICONST = 2                                                        INPU0033
C          GAUSSIAN TOLERANCE DISTRIBUTION                              INPU0034
      IDIST = 2                                                         INPU0035
C          VALUES OF C FOR SELF-CRITICAL                                INPU0036
      C(1) = 0 0D0                                                      INPU0037
      C(2) = 0 1D0                                                      INPU0038
      C(3) = 0 2D0                                                      INPU0039
      C(4) = 0 3D0                                                      INPU0040
C                                                                       INPU0041
C          INITIALIZE AVERAGES TO 0                                     INPU0042
      YBAR = 0 0                                                        INPU0043
      YSE = 0 0                                                         INPU0044
      DO 10 I = 1, NPAR                                                 INPU0045
   10 WORK(I) = 0 0                                                     INPU0046
C                                                                       INPU0047
C          LOOP OVER SAMPLE TO READ IN DATA                             INPU0048
      FLAG = NPAR  LE  2                                                INPU0049
      DO 40 I = 1, N                                                    INPU0050
        READ (IREAD,20) VC, IA(I), ALPHA, S                             INPU0051
   20   FORMAT (F10 0, I2, 2F8 0)                                       INPU0052
C                                                                       INPU0053
C          EXPRESS S IN METERS, SCALE VC                                INPU0054
        VC = ALOG(VC/1000 0)                                            INPU0055
        S = 0 3048 * EXP(S)                                             INPU0056
C                                                                       INPU0057
```

```
C        FILL THE DATA MATRIX - CONSTANT IN SECOND ROW
        X(IDEP,I) = VC
        X(ICONST,I) = 1.0
        X(3,I) = ALPHA
        X(4,I) = S
        X(5,I) = ALPHA * S
C
C        UPDATE MEAN AND STD ERROR OF DEPENDENT VARIABLE IN A
C        WAY WHICH DOESNT LOSE SIGNIFICANT FIGURES
        TEMP = VC - YBAR
        YBAR = YBAR + TEMP / FLOAT(I)
        YSE = YSE + TEMP * (VC - YBAR)
C
C        INCREMENT COVARIATE AVERAGES (3D ROW AND UP)
        IF (FLAG) GO TO 40
        DO 30 J = 3, NPAR
   30   WORK(J) = WORK(J) + X(J,I)
C
   40 CONTINUE
C
C        SET ISUB(*) AND ISTART(*), PROVIDING STARTING VALUES
C        FOR INTERCEPT AND SCALE
      DO 50 I = 1, NPAR
        ISUB(I) = I
        ISTART(I) = 0
   50 CONTINUE
      TEMP = FLOAT(N)
      YSE = SQRT(YSE/TEMP)
      ISTART(IDEP) = 1
      BETA(IDEP) = DBLE(YSE)
      ISTART(ICONST) = 1
      BETA(ICONST) = DBLE(YBAR)
C
C        WRITE OUT INFORMATION ON STD. ERROR AND AVERAGES
      WRITE (IPRINT,60) YBAR, YSE
   60 FORMAT ('1DEPENDENT (STRESS) VARIABLE'/'OMEAN =', E14.6,
     1        ' STANDARD DEVIATION =', E14.6)
      IF (FLAG) RETURN
      WRITE (IPRINT,70)
   70 FORMAT ('OCOVARIATES HAVE BEEN CENTERED BY THEIR MEANS'/
     1        'OVARIABLE', 10X, 'MEAN')
      DO 90 I = 3, NPAR
        WORK(I) = WORK(I) / TEMP
        WRITE (IPRINT,80) ISUB(I), WORK(I)
   80   FORMAT ('O', I8, E14.6)
   90 CONTINUE
C
C        SUBTRACT MEANS FROM COVARIATES
      DO 110 I = 1, N
        DO 100 J = 3, NPAR
  100   X(J,I) = X(J,I) - WORK(J)
  110 CONTINUE
C
      RETURN
      END
```

MASTER COPY - FOR REPRODUCTION PURPOSES

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A178 545

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ARO-3	2. GOVT ACCESSION NO. N/A	3. RECIPIENT'S CATALOG NUMBER N/A
4. TITLE (and Subtitle) An Algorithm for the Computation of Generalized Likelihood or Self-Critical Estimators for Binary Data		5. TYPE OF REPORT & PERIOD COVERED Working Paper
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) T.A. Delehanty A.S. Paulson		8. CONTRACT OR GRANT NUMBER(s) DAAG29-81-K-0110
9. PERFORMING ORGANIZATION NAME AND ADDRESS Rensselaer Polytechnic Institute Troy, New York 12180		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office Post Office Box 12211 Research Triangle Park, NC 27709		12. REPORT DATE
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA

18. SUPPLEMENTARY NOTES

The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

binary data, extreme value distribution, logistic distribution, Gaussian distribution, generalized likelihood, model-critical analysis, regression models, parametric proportional hazard models

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper describes the computational algorithms for computing generalized likelihood estimators for parametric proportional hazards models.

END

5-87

DTIC